Study of Transverse Wave Propagation in PMMA Beams with Adhesive Joints

A Thesis Submitted
in Partial Fulfilment of the Requirements
for the Degree of

MASTER OF TECHNOLOGY

by

SIMANT RANJAN UPRETI

to the

DEPARTMENT OF CHEMICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

DECEMBER, 1994

Certificate

This is to certify that the work contained in this thesis, entitled *Study of Transverse wave Propagation in PMMA Beams with Adhesive Joints*, has been carried out by *Simant Ranjan Upreti* under my supervision and that this work has not been submitted elsewhere for a degree.

Dr. Anil Kumar
Professor
Department of Chemical Engineering
Indian Institute of Technology
Kanpur, INDIA

December 23, 1994

Abstract

Ultra-thin joints in polymethylmethacrylate (PMMA) beams of uniform rectangular and tapered cross-section were introduced by joining the pieces using 1,2-dichloroethane. The property of the joint was varied using two modifiers — (1) dioctylphthalate plasticiser and (2) non-solvent cyclohexane. Flexural waves in the beam were propagated by impacting the beam with a small steel ball as well as with a calibrated impact hammer. The flexural response is found to embody significant parameters, in frequency and time domain, which are sensitive to the variation in the modifier content of the joint in a beam. The acceleration at a given point on the beam is assumed to be that due to forward wave, wave coming from the ends of the beam and that of the wave emerging from the joint. An optimal search technique using Genetic algorithms is developed and used to separate the wave, corresponding to the joint, which is shown to carry information (the modifier content) about it. It appears that the technique proposed in this work can be successfully used to characterise the mechanical properties of the joint.

to

my parents

Acknowledgement

Profound gratitude I express to Dr. Anil Kumar who, being a perennial source of afflatus and positivism, has assiduously nurtured the research acumen in me. Very much indebted I am to Dr. Sudhir Kamle for his alacrity in introducing the experimental techniques indispensable to this study.

I revere Dr. Santosh K. Gupta for meticulously ensconcing the quintessential computational techniques in me. I prize them. I am thankful to Dr. Kalyanmoy Deb for initiating me in Genetic Algorithms which sped my work. I am indebted to Mr. Subba Rao for helping me with *GPIB*.

All of my friends have provided me with a beneficial distraction during my work. For example, wavelets and chaos of Chaveli, LaTeXand fractals of Nandi, *Unix* and *Xfig* of Shafi, rock music of Chuphal, astrology and classical music of Presi, flute of Ashish, guitar of Sunder and Remsang, *DSP* of Balvinder and boisterous mirth of Seth, Neeraj, Sibes and Sahu have very much rubbed on me.

Last but not least, I thank my labmates Alurkar, Sailaja, Samar and Sanjay for being so congenial.

Simant

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The use of adhesive bonds in primary structures is restricted due to limited growth of methodologies for predicting their life. Inspite of this limitation, it is preferred because of the distinct advantages they offer such as uniform stress distribution, sealing against corrosion, damage tolerance and stealthiness. It is desired to characterise the properties of adhesive bonds and to establish quantitative relations between the materials's structures and its mechanical performance[1–5]. Among the existing techniques, the most commonly used are pulse echo, through transmission, resonance spectroscopy, etc. All of these techniques proceed in two stages — (a) in the first stage, known as forward problem, wave speed is measured for given joint properties while (b) in the second stage, known as reverse problem, wave speed is measured through joints to find out the joint properties.

An important characteristic of adhesive joints is that there is a definite limit to the bond strength developed between specified adherends for a particular adhesive[6–10]. With double overlap joints the load bearing capacity increases with increase in overlap until a definite value is attained. No greater load transfer can be obtained beyond this value. Experiments have shown that load transfer is mostly confined to the two end zones with a lightly loaded elastic trough in between. Another inherent characteristic of adhesive joints is that the maximum potential bond shear strength is

determined by the adhesive strain energy in shear per unit bond area. This constitutes a very important criterion. The precise shape of the stress-strain curve has been shown to have little effect on the limiting joint strength. For common adherend materials, the charts of joint efficiency are available. Since the adhesive films behave differently under peel and shear, the results of these charts are sometimes misleading. For moderate thicknesses the limiting shear strength of the adhesive may limit the joint strength. Contrary to this, for thicker adherends, the limiting strength is invariably the peel strength.

Some of the major parameters that affect adhesive bonds are bondline thickness, surface roughness, chemical conditions, adhesive type, environmental conditions and aging. When an adhesive joint fails, failure occurs either because of the failure of adhesive holding together the adherends or because of the failure of the interfaces between the adhesive and the adherends[11–21]. Normally, a joint fails with a combination of both of these. The former is called cohesion failure and is normally detectible with resonance spectroscpopy. The latter is called adhesion failure the detection of which, through the conventional techniques, is a difficult task. The traditional ultrasonic techniques have not been successful so far in this regard. The adhesive and adherends could appear in a physically perfect bonding state and yet may lack the expected strength. Microscopic examination of failed joints reveals that the failure initiates at the microvoids created by the air entrapped or by the reaction products released during the curing of adhesive. Work has been started to understand the propagation of waves at the bond interface. The properties of adhesive joints also depend upon the nature of the interface, the adhesive, the materials to be joined and the age.

To characterise the properties of these joints, various theoretical models have come up[22–29]. They can be classified into complete bond thickness model and interface weakness model. The former analyses the propagation of waves, such as

plate waves, leaky waves, interface waves, etc., through the adhesive layer and the adherends. The signals received across are found to be affected both by the interface as well as by the cohesive weakness. The interface weakness model evaluates the interface weakness by examining the transmission factors of the interface between the adhesive and the adherends. An interface wave which has smooth contact with the adherends and the adhesive and which acts as a viscous liquid layer has been shown to exist. In another quasi-static model of a weak bond, a spring with an elastic constant and mass is assumed to connect two solid semi-infinite spaces. Fracture mechanics approach has been widely used to correlate the quasi-static and fatigue crack growth behaviour of adhesive joints[30-47].

In all of these models, different bonding qualities pertain to different bonding conditions such as debonding, kissing bonding, smooth or slip bonding, rigid or welded or perfect bonding. Debonding, which comprises of formation of voids and complete separation of adhesive and adherends, can be normally detected with pulse echo through transmission, C scan, etc. The kissing bonding assumes that the traction force and particle velocity are continuous only in the direction normal to the interface and that the tangential component of the traction force is zero. Unfortunately, the analytical reflection and refraction factors from kissing bonds come out to be frequency independent, thus, negating the experimental findings. Of primary concern is smooth bonding as it can be easily missed with a conventional normal beam longitudinal wave inspection procedure. Smooth bonding can be easily simulated experimentally with a thin film between two solid media. Rigid bonding is considered as a perfect contact between two media and it assumes the continuity of traction force and particle velocity at the interface.

In the present work, 1,2 - dichloroethane (DCE) has been used to join two peices of polymethylmethacrylate (PMMA) to form beams of two different geometries :

(1) uniform rectangular and

(2) tapered.

The nature of the joint has been changed by adding separately :

(a) plasticiser dioctylphthalate (DOP) and

(b) non-solvent cyclohexane (CH)

to DCE. Addition of (a) changes the viscoelastic nature of the joint while that of (b) changes the nature of the joint-interface. Flexural waves, propagated in the beam, generated through impacting the same, are employed to characterise property of the adhesive joint. Experiments are conducted on polymethylmethacrylate (PMMA) beam because it is easy to change the property of its joints.

Analysis of the flexural responses of beam in spectral as well as time domain shows existence of seemingly important parameters sensitive to the property of the joint. It is well known that the flexural wave at any location is a superposition of a forward wave, a wave bearing information of joint and a wave due to echoes from the ends of the beam. Against this backdrop, a suitable search and optimisation technique is developed and used to resolve, the composite flexural wave into its components. The wave corresponding to the joint is found to be sensitive to the variation in the viscoelastic as well as the interfacial property of the joint and the experiments, conducted, demonstrate a correlation that exists between them.

Chapter 2

Experimental Set-up

As illustrated in Figure 1, a beam of polymethylmethacrylate (PMMA) was suspended free, horizontally, with two threads tied to its ends from a rigid steel frame. Flexural waves were generated by suitably impacting the beam at its centre. Quartz piezotron accelerometers were used to pick up the acceleration response of the beam from different locations every 3.90625 microsecond. A multi-channel signal coupler was connected to the accelerometers as well as to the impact hammer to condition their response commensurate with a Fast Fourier Transform (FFT) Analyser interfaced to the computer at the receiving end.

2.1 Joint Preparation

The polymethylmethacrylate (PMMA) beams of two different geometry, (1) uniform rectangular and (2) tapered, as shown in Figure 1a and Figure 1b were cut as shown in Figure 1c and joined by a butt joint at a distance of 30 cm from one end using 1,2-dichloroethane (DCE) solvent. This technique of joining PMMA is known to give excellent adhesion of PMMA sheets. DCE is an exceedingly good solvent of PMMA material and also has a very high vapour pressure because of which it quickly leaves the joint through evaporation. The advantage of joining PMMA pieces this way is that the thickness of the adhesive layer is close to zero and the properties of

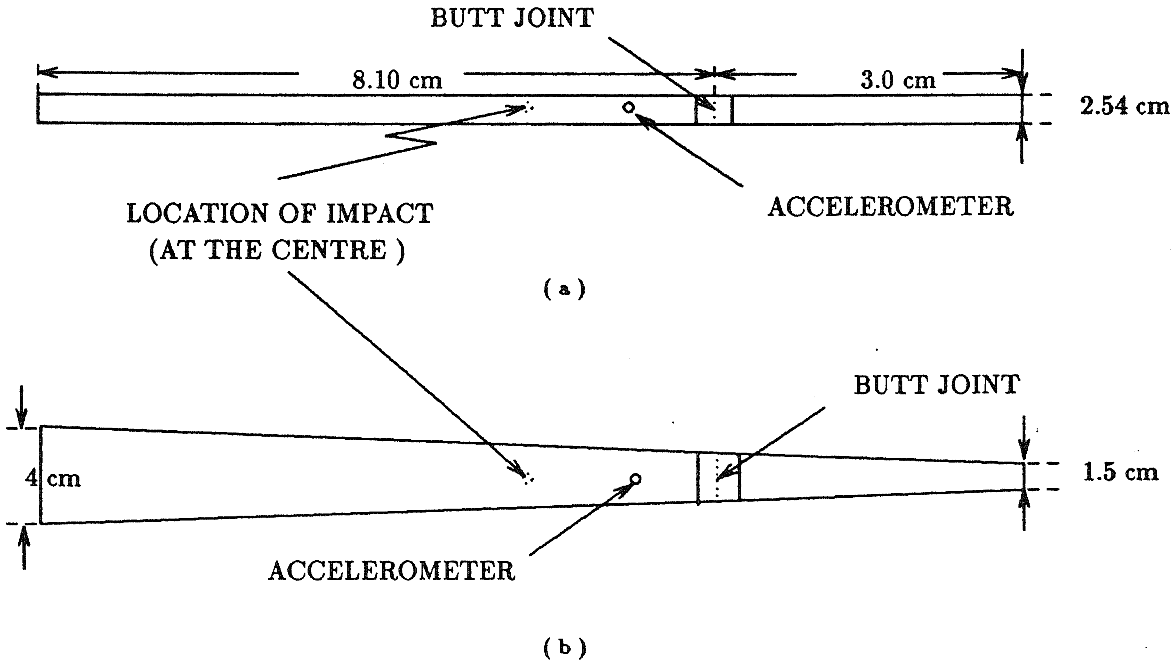
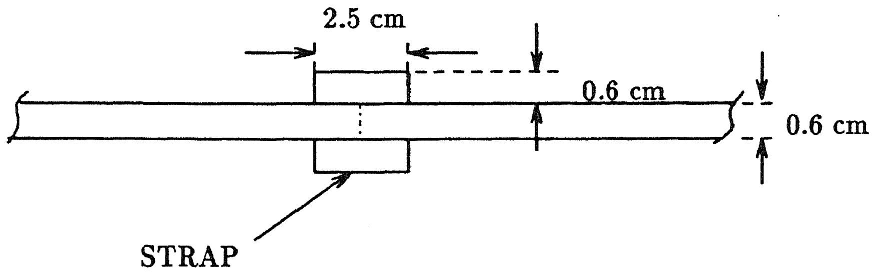
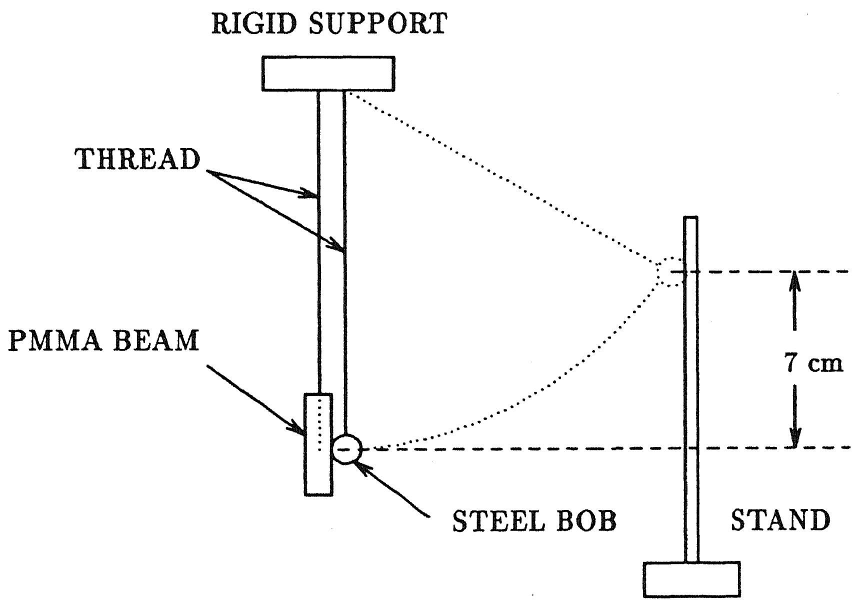


Figure 1. The PMMA beam — a) rectangular, b) tapered, c) butt joint d) under impact.



(c)



(d)

Figure 1. The PMMA beam — a) rectangular, b) tapered, c) butt joint d) under impact.

the interfacial region are identical to those of the PMMA beam. In order to vary the property of the joint, two different modifiers, plasticiser dioctylphthalate (DOP) and non-solvent cyclohexane (CH), were mixed separately with DCE. DOP varies the viscoelastic properties of the joint, while CH varies the interfacial property of the same. Upon evaporation of the solvent, the modifier, DOP or CH, is left behind at the interface which changes either its viscoelastic or interfacial characteristics.

2.2 Generation of Flexural Waves

Flexural waves were generated in the beam by striking it in the middle with a steel ball, 10 mm in diameter, from a fixed height of 70 mm as shown in Figure 1d. An accelerometer was positioned on the beam at a distance of 4.30 cm from the point of impact towards the joint. A computer program was developed using General Purpose Interface Bus (GPIB) to interface FFT Analyser to a Personal Computer (refer Appendix A). This ensured efficient experimentation yielding high fidelity flexural response data. On the basis of visual inspection of the response, it was possible to isolate a portion from the flexural response curve (of 20 msec duration) and to analyse it for different types of joints. The demarcation of this portion is given in Table 2.1. The results of the analysis show the existence of a significant parameter sensitive to the joint property. Further experimentations were performed with a calibrated impact hammer.

2.3 Flexural Wave Generation using Calibrated Impact Hammer

A calibrated impact hammer was hung free from the frame and was used to apply controlled force to generate flexural waves in the beam. A slotted angle on a support was used to guide the impact hammer from a given height and distance from the beam. The relative motion between the thread and the beam and that between

Table 2.1: Approximate division of flexural response generated by steel ball

Beam Geometry	*Uniform Rectangular*
Station	*4.3 cm from the Point of Impact*
Modifier	*DOP Plasticiser*
Time Frame	*20 millisecond*

Section	Sampling Instants
1	50-230
2	231-440
3	441-650
4	651-1023

the iron frame was eliminated by properly gluing their points of contacts. This obviates any friction which might interfere with the propagation of flexural waves in the beam. Flexural responses were recorded for a period of 4 millisecond during which the hammer impacted only once. A small PMMA point of negligible mass was attached on the point of impact at the centre of the beam in order to spatially localise the impact force. The peak value of the imulse force was ensured to lie within $\pm 5\%$ of a specified value. This gave the best reproducible results.

Chapter 3

Theoretical Development

It may be recalled that any signal $g(t)$ satisfying Dirichlet's conditions for finite duration can be expressed with the help of *fourier integral* [48] as

$$g(t) = \int_{-\infty}^{+\infty} G(f) exp\,(j2\pi ft)\, df \tag{3.1}$$

where $G(f)$ is defined by the *fourier transform*

$$G(f) = \int_{-\infty}^{+\infty} g(t) exp\,(-j2\pi ft)\, dt. \tag{3.2}$$

For the problem of one dimensional transverse wave propagation in a PMMA beam, it is assumed that at any given location on a beam, the flexural response is a superposition of three types of waves — a) forward wave, $g_f(t)$, arising from the impacting of the beam, b) echo, $g_j(t)$, from the joint and c) echo, $g_e(t)$, from the ends of the beam. This implies that

$$g(t) = g_f(t) + g_j(t) + g_e(t). \tag{3.3}$$

In terms of its Fourier transform, this equation can be written as

$$g(t) = \int_{-\infty}^{+\infty} [G_f(f) + G_j(f) + G_e(f)]\, exp\,(j2\pi ft)\, df. \tag{3.4}$$

It is desired to separate the waveform corresponding to the joint because it is expected that it should carry some information concerning the nature of the joint. To

do so, Eqn. (3.4) is discretised as follows treating $G_f(kF_s), G_j(kF_s)$ and $G_e(kF_s)$ as complex variables, to be optimally determined later :

$$g(nT_s) = \frac{1}{NT_s}\sum_{k=0}^{L-1}\left[\mathcal{G}_f(kF_s)+\mathcal{G}_j(kF_s)+\mathcal{G}_e(kF_s)\right] exp\left(j\frac{2\pi}{N}kn\right), \quad (3.5)$$
$$L < N,\ n = 0, 1, ..., L-1.$$

Above, $g(nT_s)$ represents a finite sequence of N samples of the flexural response separated in time space by T_s instants where $1/T_s$ is greater than or equal to twice the highest frequency component of the signal. The coefficients $\mathcal{G}_f(kF_S), \mathcal{G}_j(kF_s)$ and $\mathcal{G}_e(kF_s)$ are complex finite sequences of L samples separated in the spectral space by F_s hertz.

3.1 The Optimisation Problem

The optimal complex coefficients, $\mathcal{G}_f(kF_s)$, $\mathcal{G}_j(kF_s)$ and $\mathcal{G}_e(kF_s)$, are determined so that the response $g(t)$ in Eqn. (3.3) minimises the sum of the squares of difference between a) the response reconstructed from these three coefficients and b) the actual response in time space. The optimisation problem is, thus, mathematically stated as

$$I = \min_{\mathcal{G}_f(kF_s),\mathcal{G}_j(kF_s),\mathcal{G}_e(kF_s)} \sum_{n=0}^{N-1}\left(\overline{g(nT_s)} - g(nT_s)\right)^2 \quad (3.6)$$

where the objective function I is the sum of the square error and $\overline{g(nT_s)}$ is the experimental acceleration value of a given point on the beam.

In the absence of joint, the acceleration at any point, $g_w(t)$, is the superposition of only $g_f(t)$ and $g_e(t)$ as

$$g_w = g_f(t) + g_e(t). \quad (3.7)$$

If in the beam with a joint, the superposition of $g_f(t)$ and $g_e(t)$ is $g_w^*(t)$, then Eqn. (3.3) can be rewritten as

$$g(t) = g_w^*(t) + g_j(t). \quad (3.8)$$

Since $g_w^*(t)$ is expected to be close to g_w, suitable guess values for $\underset{\sim}{G}_w^*(kF_s)$ and $\underset{\sim}{G}_j(kF_s)$ were chosen in order to initiate optimisation process. The guess value for $\underset{\sim}{G}_w^*(kF_s)$ was based upon the spectral coefficient of the flexural response of identical point on a beam without the joint. The difference of the corresponding spectral coefficients of the flexural responses of identical points on the beam with and without joint was used to provide the guess value for $\underset{\sim}{G}_j(kF_s)$. The choice of initial conditions in this manner is observed to preserve the pattern of the original flexural waveform.

In order to reduce the number of independent variables, L is taken to be 10, associated with the first ten spectral points with $N = 1024$ in time space. For this, the problem consists of optimization in a forty variable (both real and imaginary) search space. With flexural wave data collected so far, this value of L has been observed to be a good compromise between the desirable accuracy of result and the computer time.

3.2 Genetic Algorithms

Emulating the principles of evolution of life on earth, Genetic algorithms (GAs) are robust and versatile optimisation techniques [49,50]. These do not require the objective function to be continuous and/or differentiable and are insensitive to starting conditions. The basic structure of a standard GA is shown in Figure 2 and its working is demonstrated with a simple example. As shown in the flow chart, the guesses leading to higher values of objective functions are favoured with the help of Monte-Carlo selection technique. For the example shown in Figure 2, the Monte-Carlo selection gives the guesses for the next stage to be $x = 13, 20$ and 20. In Table 3.1 are defined the crossover and mutation processes which are done by writing these numbers into binary strings. These operations produce the new values of guesses which are far better than the original ones. For the example discussed, there is a local maximum at $x = 31$ and the crossover and mutation operations lead to the new guess values

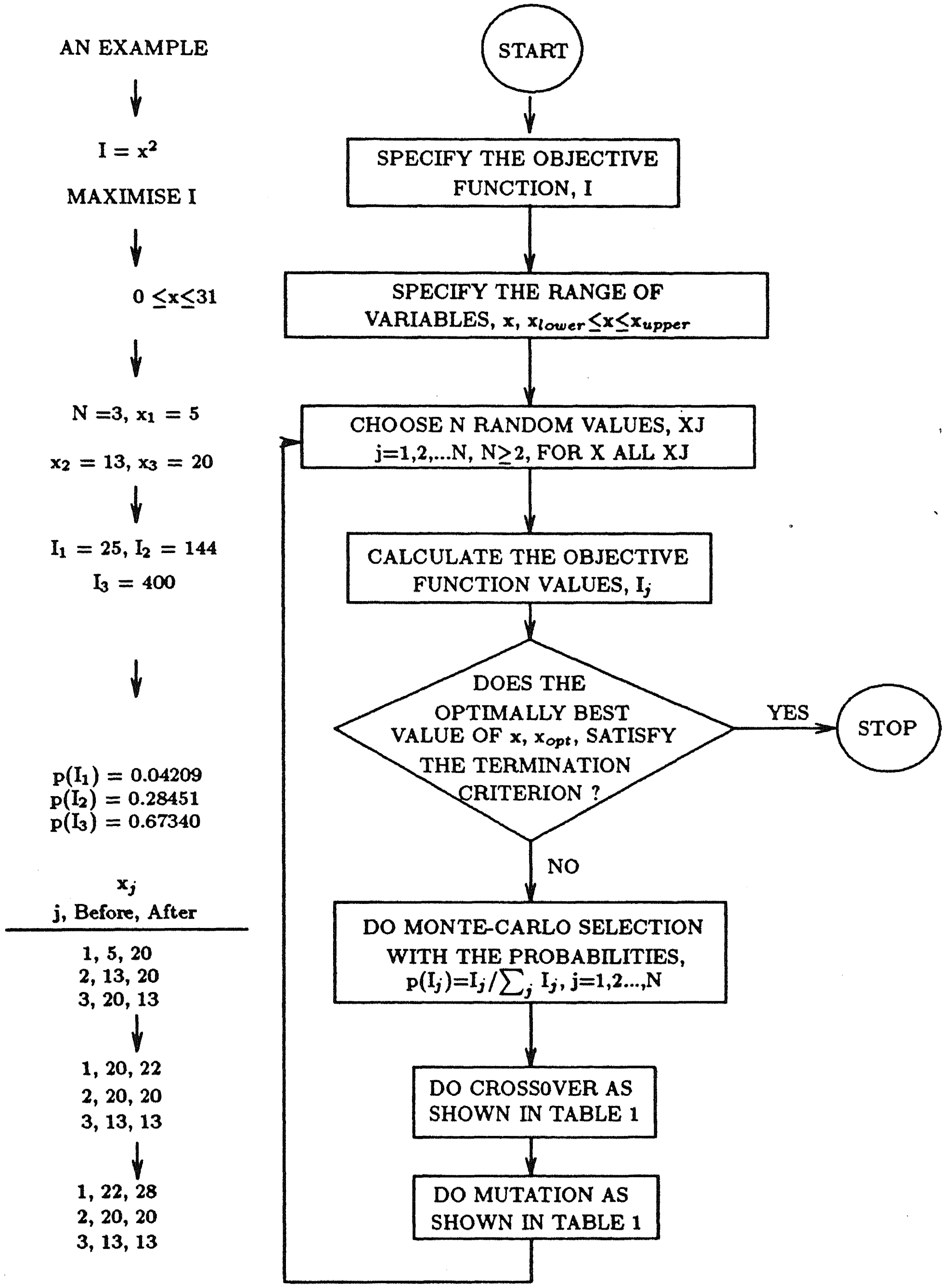


Figure 2. Structure of a standard Genetic Algorithm explained with a simple example of maximising $I = x^2$, $0 \geq x \geq 31$.

Table 3.1: The process of crossover and mutation in Figure 2

Crossover or Recombination of strings :
A string is defined as a number encoded in binary digit system. For example, $x = 20$ can be written as 1 0 1 0 0 in binary which is a string. In the process of crossover, we randomly pick two strings at a time, say $x = 13$ and $x = 20$, from a population of variable and recombine them as follows :

$$\begin{array}{ccccccc} \overbrace{(x=20)}^{y=400} & \text{or} & 101\ \vdots 00 & \longrightarrow & 10110 & \text{or} & \overbrace{x=22}^{y=484} \\ \overbrace{(x=13)}^{y=169} & \text{or} & 001\ \vdots 10 & & 00100 & \text{or} & \overbrace{x=04}^{y=16} \end{array}$$

The vertical dots, above, represent a randomly chosen crossover site, the bits right to which are swapped.

Mutation of Strings :
A string from the population of variables is picked up. With a very small probability, each bit is changed, i.e., if it is 1 then it is made 0 and if it is 0 then it is made 1 applying Monte-Carlo method. The process is repeated for all strings the in the population. For example, with the strings, $x = 22$, $x = 20$ and $x = 13$, following mutations may happen :

$$\begin{array}{ccccccc} \overbrace{(x=22)}^{y=484} & \text{or} & 01110 & & 11100 & \text{or} & \overbrace{x=28}^{y=784} \\ \overbrace{(x=20)}^{y=400} & \text{or} & 10100 & \longrightarrow & 10100 & \text{or} & \overbrace{x=20}^{y=400} \\ \overbrace{(x=13)}^{y=169} & \text{or} & 01101 & & 01000 & \text{or} & \overbrace{x=08}^{y=64} \end{array}$$

Optimally better string, $x = 28$, is substituted in place of $x = 22$ while the optimally inferior string, $x = 8$, is rejected.

---

of $x_i = 28, 20$ and 13 which are closer to the optimum value. The experience of working with this algorithm shows a fast convergence towards the optimal solution, and, particularly for the present problem, other standard search techniques do not work well enough.

The present problem consists of finding the optimal real and imaginary values of $\underline{G}_w(kf_s)$ and $\underline{G}_j(kF_s)$. This means that if L is chosen in Eqn. (3.5) to be ten, it would mean that there would be forty independent variables to optimise. A single string (defined in Table 2) is formed in which all these variables are represented as shown in Figure 3. Each slot in the string has a fixed number of binary bits. Experience in working with this has shown that randomising the initial population on the basis of guess values leads to a faster convergence. In order to do so, the random population is related to the the guess values, $\underline{G}_{w,i,0}$ and $\underline{G}_{j,i,0}$, in the following way :

$$\begin{aligned} \underline{G}_{w,i} &= \underline{G}_{w,i,0}(1 + \underline{R}_{w,i}), \\ \underline{G}_{j,i} &= \underline{G}_{j,i,0}(1 + \underline{R}_{j,i}), \\ & i = 0, 1, ..., 9. \end{aligned} \tag{3.9}$$

Here, $\underline{R}_{w,i}$ and $\underline{R}_{j,i}$ are the complex random numbers. The purpose of this formulation is to provide the inequality constraints

$$\begin{aligned} \underline{G}_{w,i,0} \leq \ \underline{G}_{w,i} \ &\leq \underline{G}_{w,i,0}(1 + \underline{R}_{w,i}), \\ \underline{G}_{j,i,0} \leq \ \underline{G}_{j,i} \ &\leq \underline{G}_{j,i,0}(1 + \underline{R}_{j,i}), \\ & i = 0, 1, ..., 9, \end{aligned} \tag{3.10}$$

if $\underline{R}_{w,i} \geq 0$ and $\underline{R}_{j,i} \geq 0$. Otherwise

$$\begin{aligned} \underline{G}_{w,i,0}(1 + \underline{R}_{w,i}) \leq \ \underline{G}_{w,i,0} \ &\leq \underline{G}_{w,i}, \\ \underline{G}_{j,i,0}(1 + \underline{R}_{j,i}) \leq \ \underline{G}_{j,i,0} \ &\leq \underline{G}_{j,i}, \\ & i = 0, 1, ..., 9. \end{aligned} \tag{3.11}$$

$Re\{\underline{R}_{w,0}\}$
$Im\{\underline{R}_{w,0}\}$
$Re\{\underline{R}_{j,0}\}$
$Im\{\underline{R}_{j,0}\}$
$Re\{\underline{R}_{w,1}\}$
$Im\{\underline{R}_{w,1}\}$
$Re\{\underline{R}_{j,1}\}$
$Im\{\underline{R}_{j,1}\}$
$\vdots$
$Re\{\underline{R}_{w,9}\}$
$Im\{\underline{R}_{w,9}\}$
$Re\{\underline{R}_{j,9}\}$
$Im\{\underline{R}_{j,9}\}$

Figure 3. Layout of a multi-variable binary string used in optimisation.

if $\underset{\sim}{R}_{w,i} \leq 0$ and $\underset{\sim}{R}_{j,i} \leq 0$. This ensures variation of the forty variable search space (by varying the order of $\underset{\sim}{R}_{w,i}$, and $\underset{\sim}{R}_{j,i}$ in the vicinity of the guess values where the optimal co-ordinates $(\hat{\underset{\sim}{G}}_{w,0}, \hat{\underset{\sim}{G}}_{w,1}, ..., \hat{\underset{\sim}{G}}_{w,9}, \hat{\underset{\sim}{G}}_{j,0}, \hat{\underset{\sim}{G}}_{j,1}, ..., \hat{\underset{\sim}{G}}_{j,9})$ are expected. Since the sum of squares of the error has to be minimised, the objective function value in the algorithm should be $1/I_i$, $i = 0, 1, ...9$. It has been taken to be $(1/I_i)^2$, $i = 0, 1, ...9$ because using the square of the objective function value has been shown converge GAs faster [51].

Chapter 4

Results and Discussions

The joints in polymethylmethacrylate (PMMA) beams of two different geometry, uniform rectangular and tapered, were prepared using solvent 1,2-dichloroethane (DCE) mixed separately with two different modifiers, plasticiser dioctylphthalate (DOP) and non-solvent cyclohexane (CH) to modify the joint properties. These modifiers have very low vapour pressures and are expected to stay near the joint after DCE is evaporated. This way one can get varied properties of the joint to be explored. Spectral analysis of the flexural responses of beams based on their certain time domain features is presented at first.

4.1 Spectral Analysis

Experiments were conducted with uniform rectangular PMMA beam with DOP as a joint modifier. Acceleration values as a function of time, of a station on a beam 4.3 centimetre after the point of impact towards the joint, were recorded after impacting the beam with a steel ball from a height of 7 centimetre. A typical acceleration response obtained is shown in Figure 4. Four discernible sections seem to exist in time domain which have different patterns, e.g., number of extrema per unit time and peak extremum value within asection. Superposing different responses, (of rectangular beams with different % DOP), these sections are marked out in

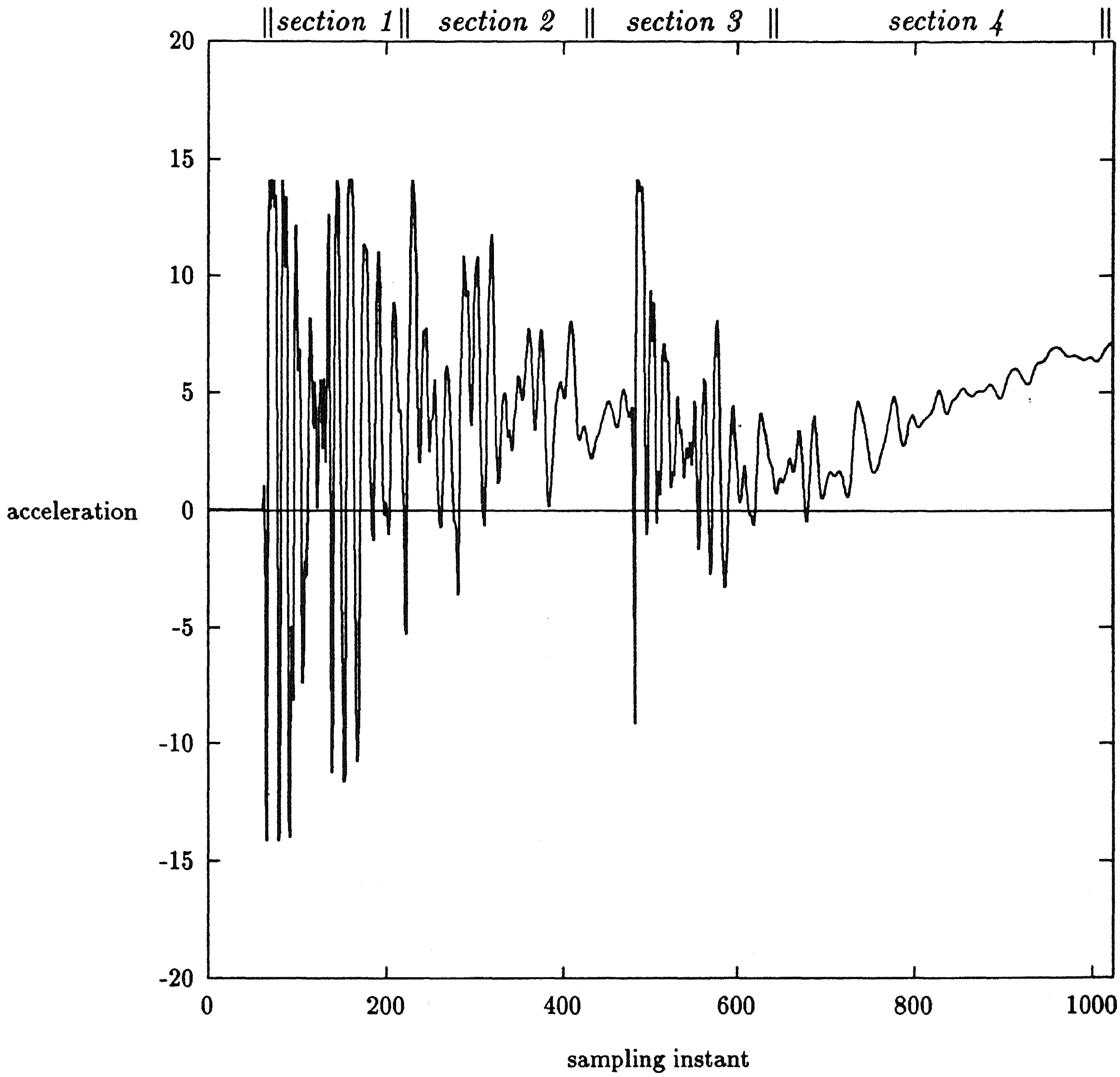


Figure 4. Acceleration versus sampling instant (1024 samples in 20 msec), at a point on the beam 4.30 cm before the joint, due to impact with a steel ball (one acceleration unit = 421.03 m/sec^2).

the figure (also refer to Table 2.1). A computer program was written to represent each section by 128 equi-spaced acceleration values. Fast fourier transforms of the time responses, of different beams, of all sections were obtained. Behaviour of the third section in spectral domain is found to be sensitive to the DOP content of the joint. Figure 5 shows the amplitude versus frequency graph of flexural responses of rectangular beams with varying amount of DOP. A careful observation of this figure shows that the second peak of the graphs is significant. In Figure 6, the amplitude of the second peak versus percentage DOP used in preparing the joints is shown. The '0% DOP' stands for an identical beam without joint. It is found that the amplitude of the second peak of the third section is a monotonically increasing function of DOP percentage. An increase of about four times in the amplitude is observed. Such sensitivity is not found for other sections and it appears that the information concerning the properties of the joint is embodied in the third section. It may be recalled that in the process of joint formation, DCE evaporates while DOP stays in a very thin region around the interface. The presence of DOP in the area of joint appears to make the small region around the interface more viscoelastic and is likely to be responsible for this effect.

One disadvantage of the experimentation with steel ball is that the frequencies of the flexural wave generated upon impacting are very large whereas sensitivity to the joint properties is higher for lower frequencies. In addition to this, multiple impacts may also occur. In an effort to understand the process of impacting, a calibrated impact hammer was employed. The force of impact as a function of time, for 20 millisecond, is presented in Figure 7. This clearly shows that there have been three significant impacts over the period of 20 milliseconds. The second and third impacts are of smaller magnitudes but cannot be really ignored or eliminated. Hence, acceleration measurements only up to 4 millisecond, during which only one controlled impact occurs, were decided to be considered. Since the impact force

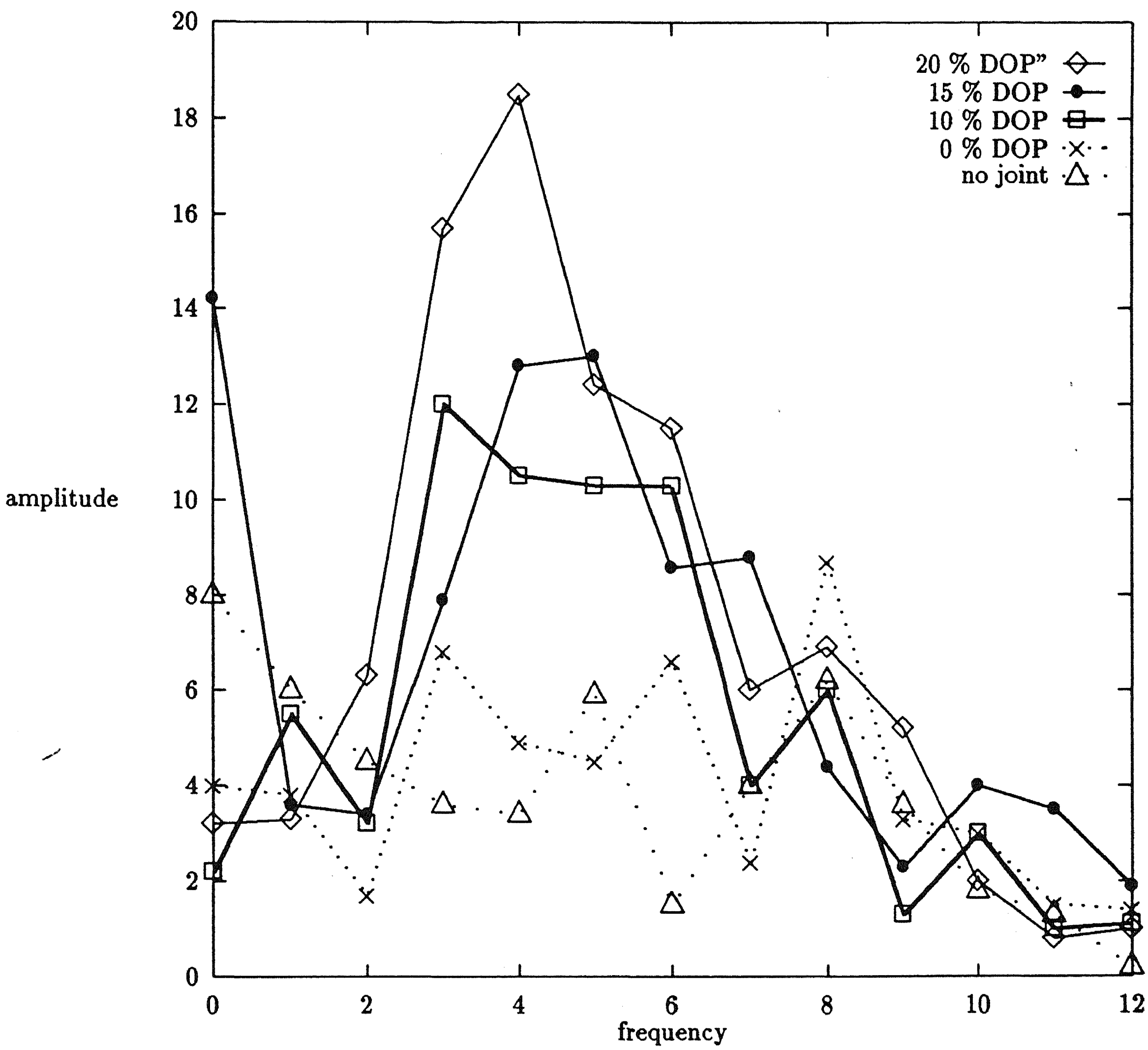


Figure 5. Amplitude versus frequency graphs (of section 3 in Figure 4) for different beams (one amplitude unit = 421.03 m/sec^2, one frequency unit = 250 Hz).

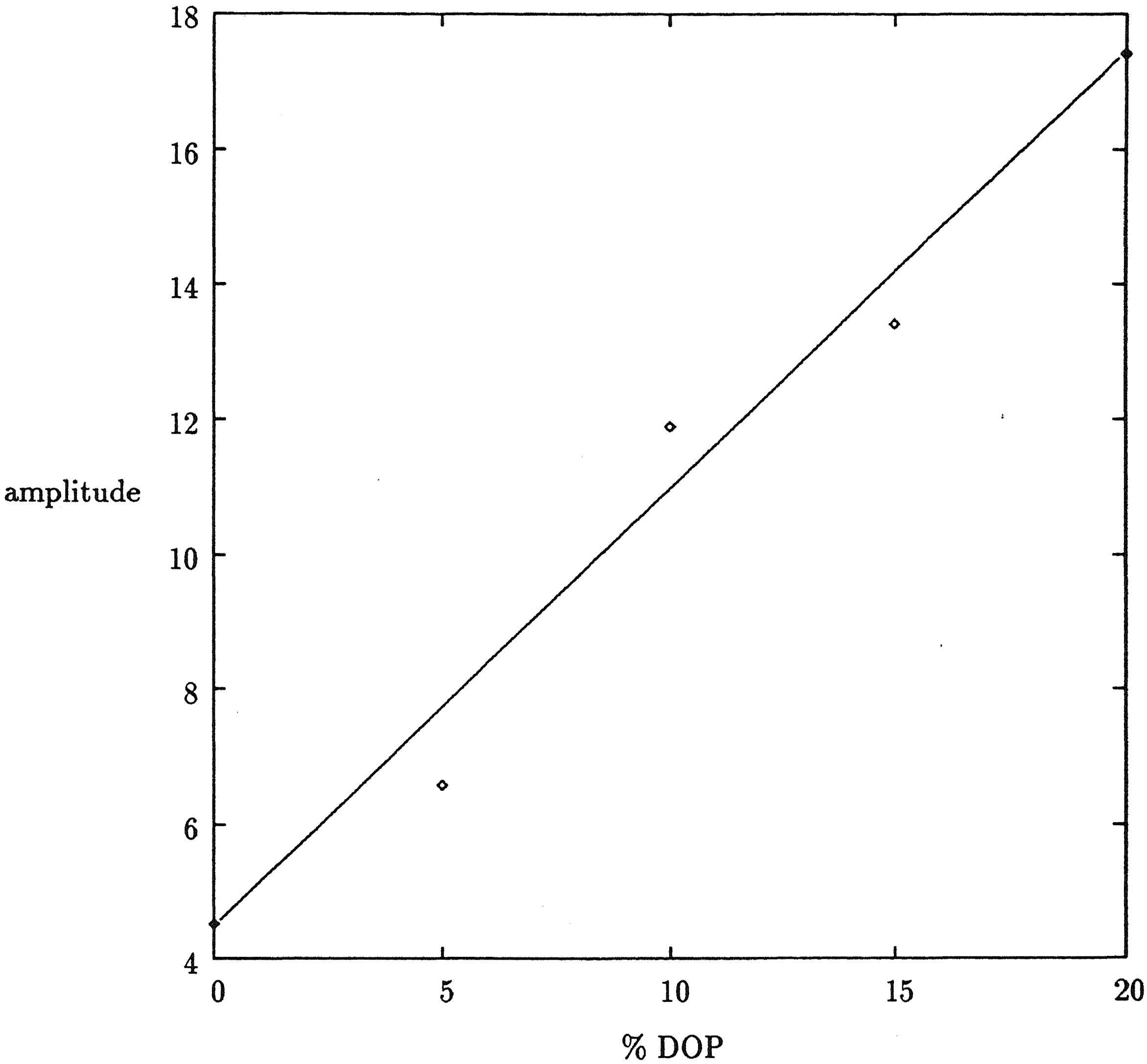


Figure 6. Amplitude of second peak (in Figure 5) versus change in DOP content of the beams (one amplitude unit = 421.03 m/sec^2).

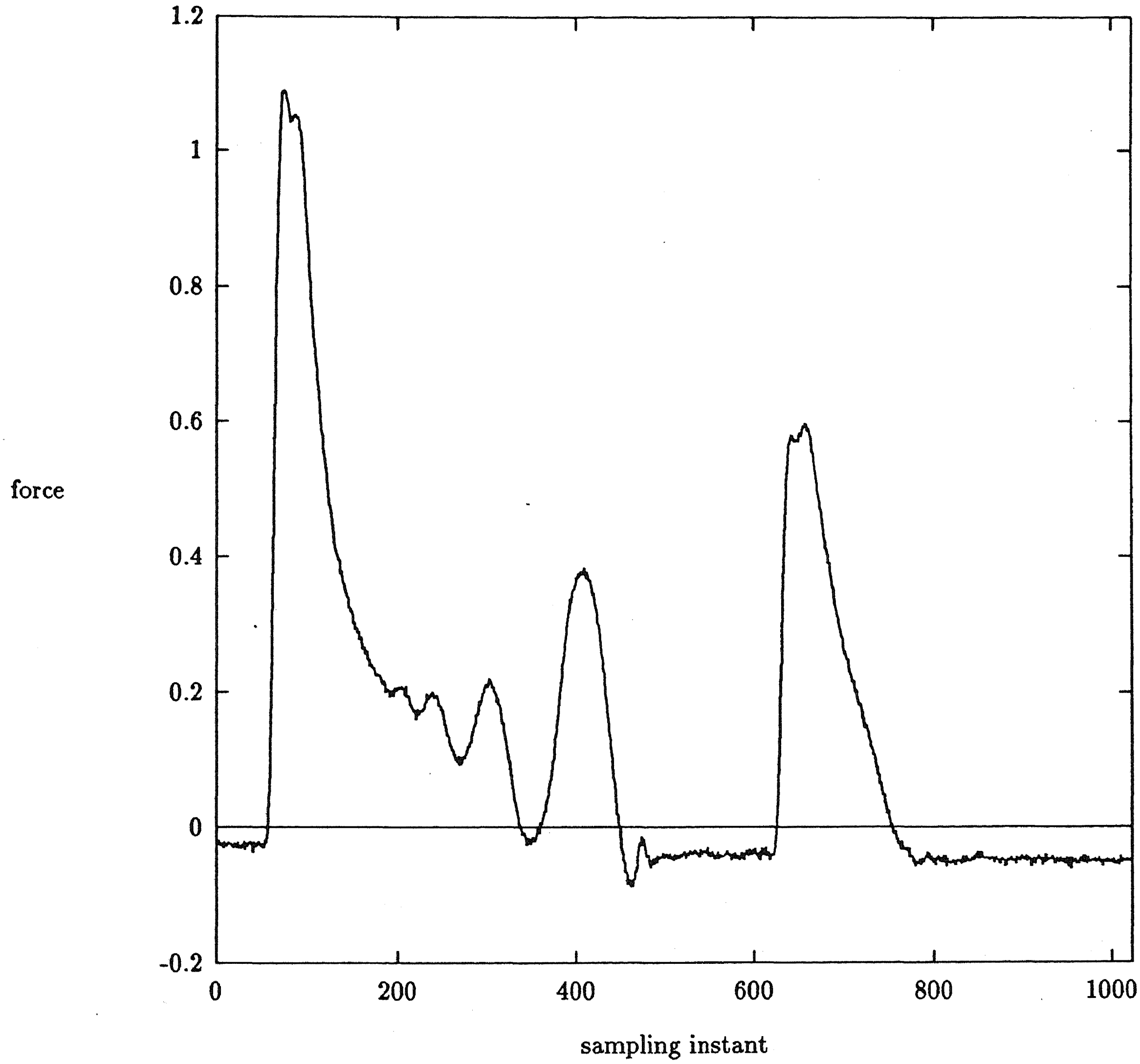


Figure 7. Force versus sampling instant (1024 samples in 20 msec), at a point on beam 5 cm before joint, due to impact with calibrated hammer (one force unit = 70.92 N).

can be measured as a function of time, the force applied can be controlled more carefully. The peak value of force was suitably restricted to within ±5% so as to have a desirable repeatability. The time domain analysis of the flexural responses, obtained after these improvisations, is described below.

4.2 Time Domain Analysis

In this study, simple butt joints, without straps, were made by joining the ends of the cut pieces of the beam. Experiments were conducted on beams with two different geometries, uniform rectangular and tapered, and with two different joint-modifiers, DOP and CH.

Figure 8 shows the responses of PMMA beam with a typical geometry and joint-modifier. It is observed that the peak value of the flexural response of any beam decreases significantly with the amount of modifier present in the joint. Figure 9 exihibits this result for beams with different geometry and modifier. The following observations are patent from Figure 9 :

1. The rate of decrease of peak value is higher with CH than with DOP irrespective of the geometry of the beam. In other words, the peak value is more sensitive to CH than DOP content of the joint.

2. The rate of decrease of peak amplitude with a given modifier, virtually, does not vary with change in beam geometry.

3. With same joint-modifier, the peak values of tapered beam are always higher than that of rectangular beam for a given amount of modifier present in the joint.

Besides this, the flexural data of the standardised experiments were used to find out the properties of the wave, $g_j(t)$, corresponding to the joint. The optimisation technique developed above was employed for this purpose.

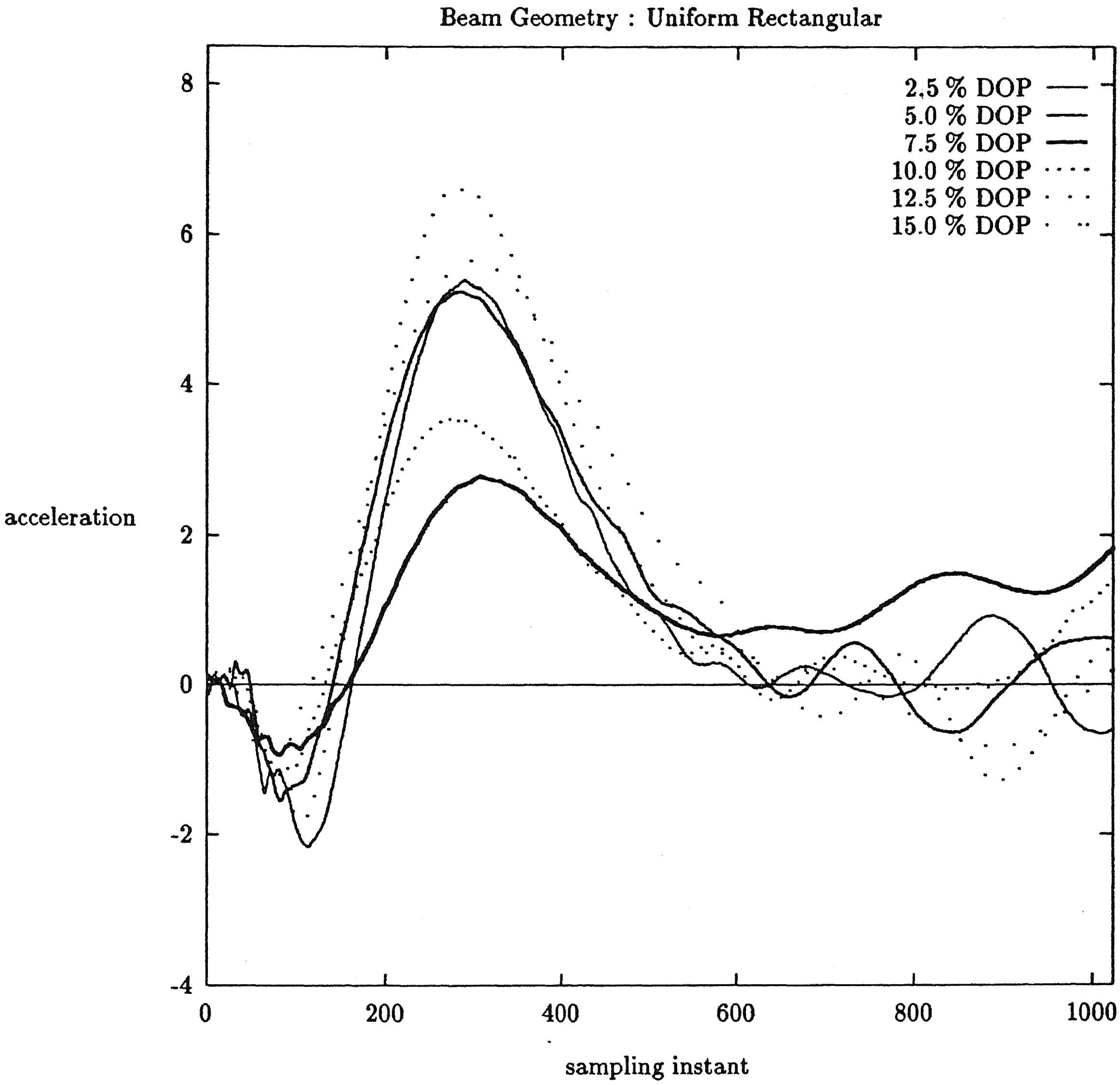


Figure 8. Flexural response of PMMA beams with different geometry and modifier.

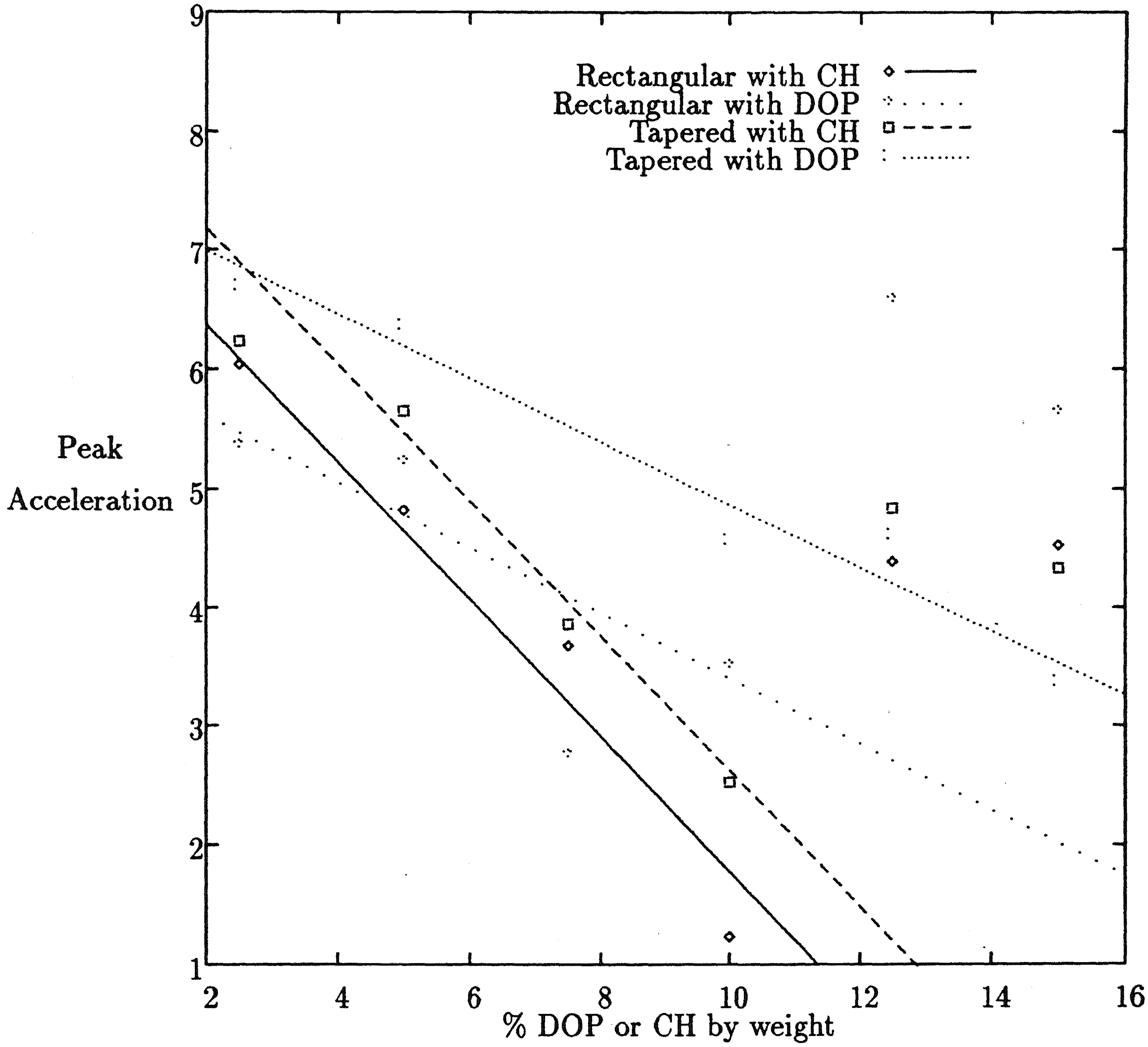


Figure 9. Peak values of flexural response of PMMA beams versus modifier content with different geometry and modifier.

4.3 Extraction of the Joint-Wave

The wave $g_j(t)$ gets separated from the composite flexural response of the beam when the objective function I in Eqn. (3.6) is minimised. A computer program was developed to effect the same (refer Appendix B). Figure 10 shows the values of the objective function versus the iteration number, in the Genetic algorithm, for flexural responses obtained before and after the joint on a typical beam. It is interesting to note that I rapidly reduces from $O(10^2)$ to a satisfactory $O(10)$. For the typical responses, the acceleration profile reconstructed from the optimal coefficients, $\hat{\underline{G}}_w(kF_s)$ and $\hat{\underline{G}}_j(kF_s)$, is always found to agree very well with the experimental acceleration profile. Figure 11 illustrates how the response reconstructed from these optimal coefficients converges to the experimental flexural response with increase in the number of iterations of the optimisation algorithm developed above.

Table 4.1 gives the computed results of the real and imaginary amplitudes for the first two frequencies, $k = 0$ (500π rad/sec) and $k = 1$ (750π rad/sec) for beams with different geometry and modifier. It is noticed that, for any beam of a given geometry, only the optimal values of $\hat{\underline{G}}_w(kF_s)$ remain virtually the same as their corresponding initial guess values, $\underline{G}^*_w(kF_s)$. This suggests that $\hat{\underline{G}}^*_w(kF_s)$ solely depend upon the geometry of the beams. The optimal coefficients, $\hat{\underline{G}}_j(kF_s)$ are found to be sensitive to the change in modifier content of the joint and, thus, should carry information about joint.

With different random seeds, used to generate random numbers $\underline{R}_w$ and $\underline{R}_j$ of Eqn (9), the optimal coefficients, $\hat{\underline{G}}_w(kF_s)$ and $\hat{\underline{G}}_j(kF_s)$ are found to be virtually the same for any flexural response. This suggests that the result obtained represent the true optimal values. For a typical beam, Table 4.2, illustrates this point.

Since the results of Table 4.1 do not show any discernible correlation between the amplitude of the optimal coefficients corresponding to joints and the modifier content, efforts were made to explore the changes occuring in the transmission of

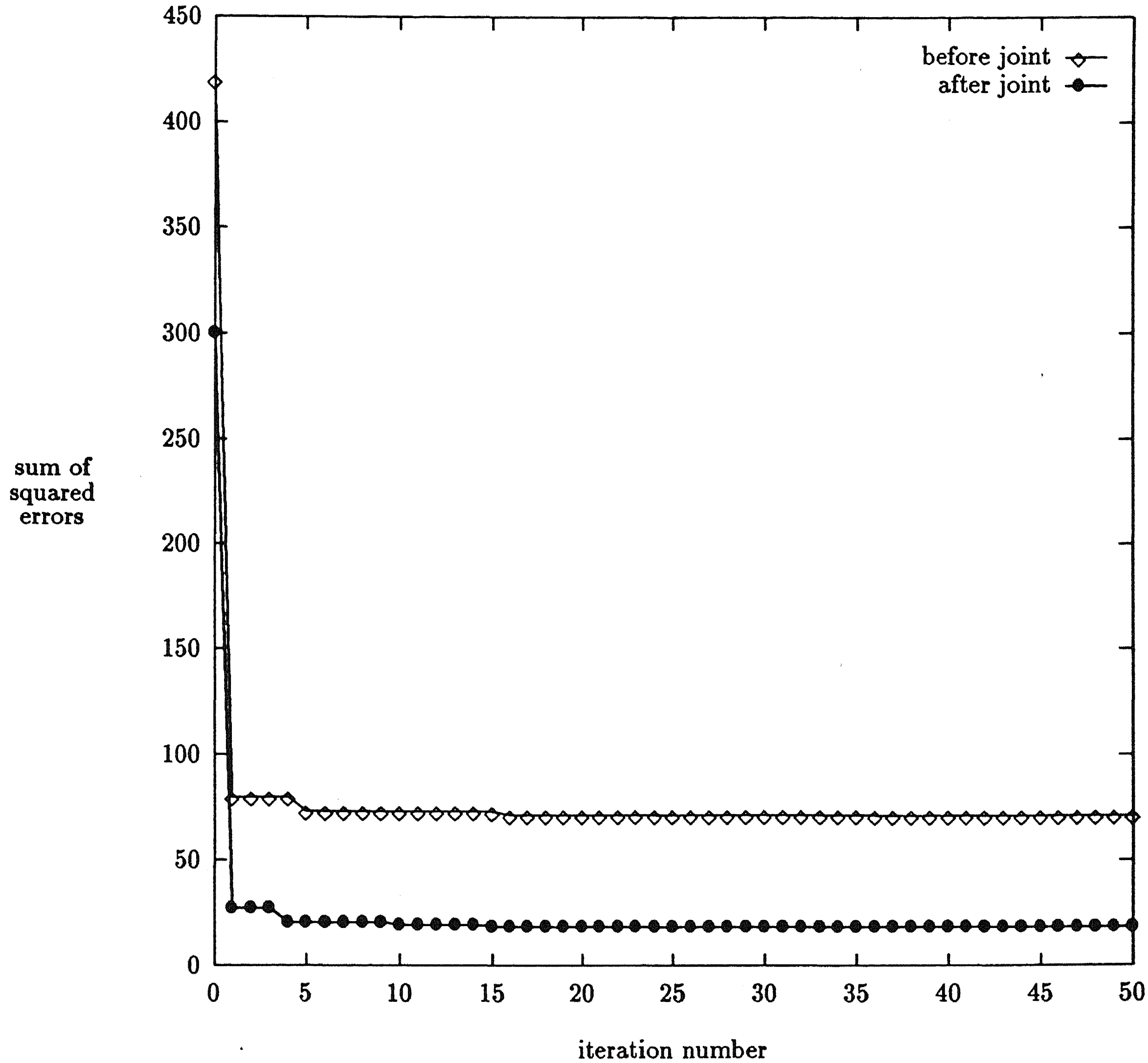


Figure 10. Sum of squared errors versus iteration number, of the optimisation code, for a typical beam.

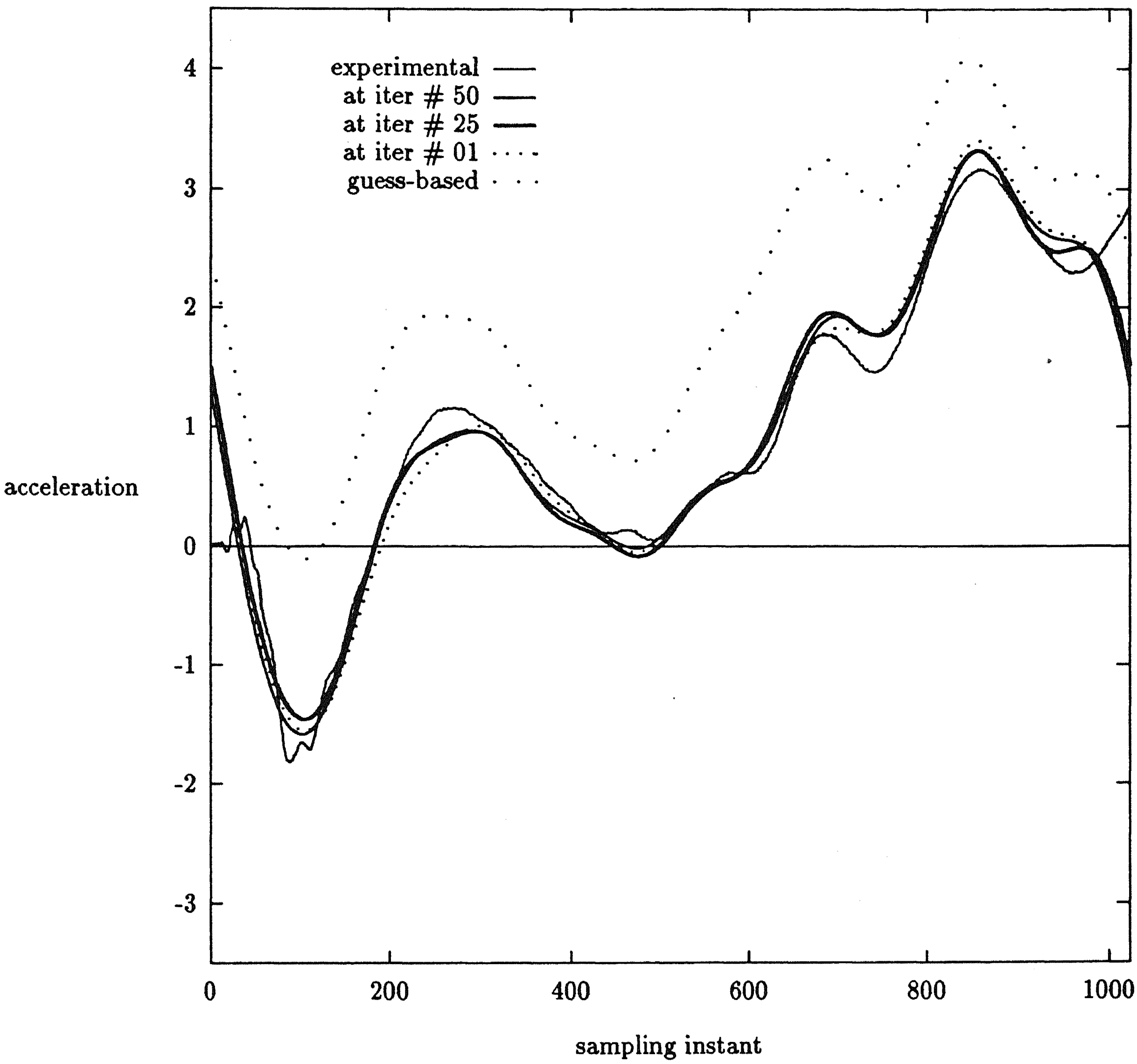


Figure 11. Convergence of the guess-based profiles to the corresponding experimental ones (1024 samples in 4 msec, one acceleration unit = 421.03 m/sec^2).

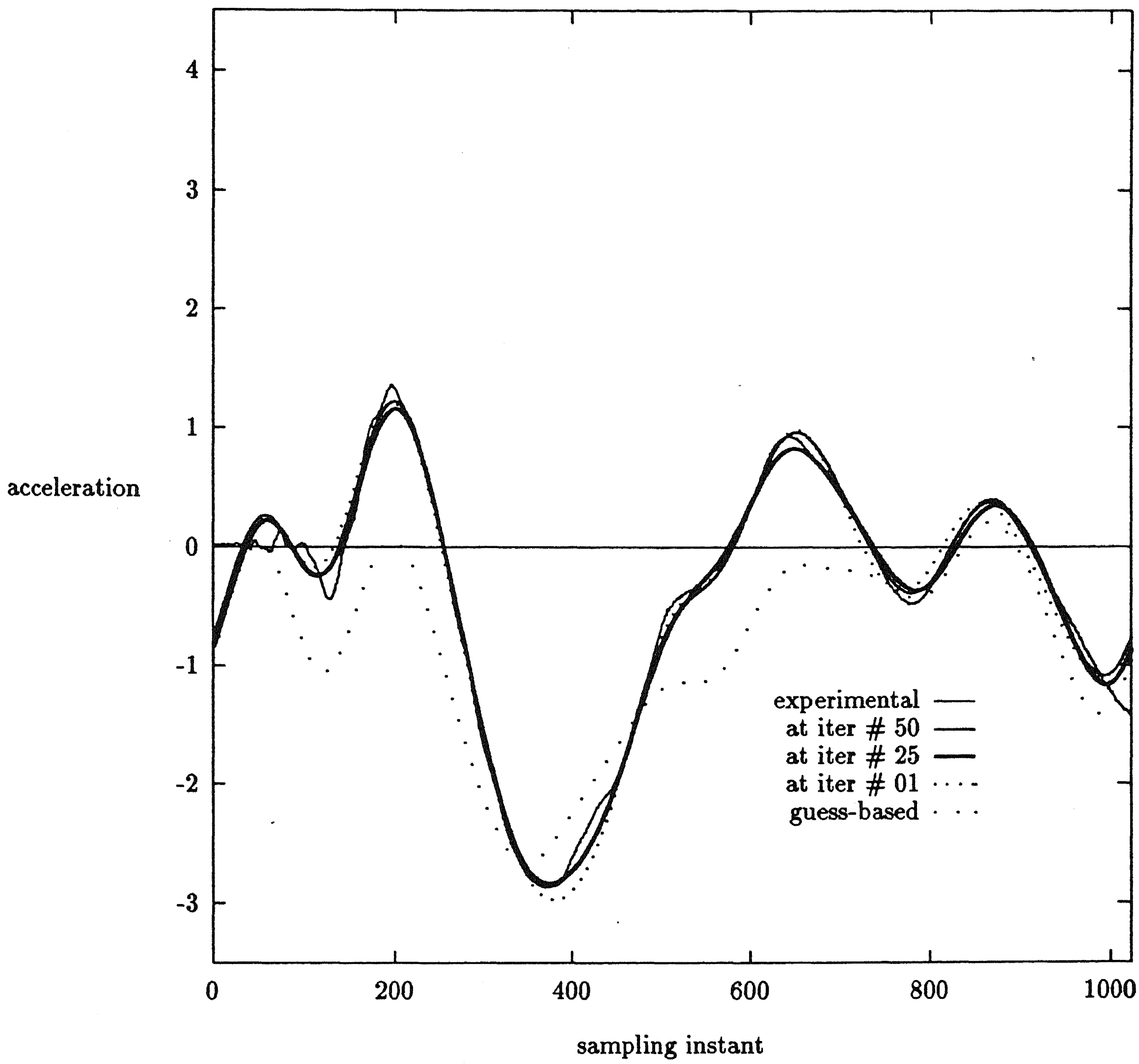


Figure 11. Convergence of the guess-based profiles to the corresponding experimental ones (1024 samples in 4 msec, one acceleration unit = 421.03 m/sec^2).

Table 4.1: Initial and optimal values of complex coefficients
Beam Geometry Rectangular, *Modifier* CH, *Location* Station 5 cm before Joint.

$k=0$ % DOP		$Re\{G_{w,0}\}$	$Re\{G_{j,0}\}$	$Im\{G_{w,0}\}$	$Im\{G_{j,0}\}$
2.5	initial	1195.753	-251.038	.0	.0
	optimal	1231.625	-258.569	.0	.0
5.0	initial	1195.753	-9.80298	.0	.0
	optimal	1207.710	-9.90101	.0	.0
7.5	initial	1195.753	-406.723	.0	.0
	optimal	1243.583	-422.992	.0	.0
10.0	initial	1195.753	-2697.9	.0	.0
	optimal	1195.753	-2697.9	.0	.0
12.5	initial	1195.753	-343.045	.0	.0
	optimal	1207.710	-346.475	.0	.0
15.0	initial	1195.753	-136.526	.0	.0
	optimal	1195.753	-136.526	.0	.0

$k=1$ % DOP		$Re\{G_{w,1}\}$	$Re\{G_{j,1}\}$	$Im\{G_{w,1}\}$	$Im\{G_{j,1}\}$
2.5	initial	89.14785	-471.485	-863.594	-178.276
	optimal	175.6212	-928.825	-1714.23	-377.945
5.0	initial	89.14785	-611.322	-863.594	79.67595
	optimal	177.4042	-1216.53	-1730.64	169.7097
7.5	initial	89.14785	-240.693	-863.594	434.484
	optimal	179.1871	-483.793	-1779.87	938.4854
10.0	initial	89.14785	83.61193	-863.594	343.5609
	optimal	181.8616	170.5683	-1821.32	752.3983
12.5	initial	89.14785	-384.342	-863.594	369.35
	optimal	160.4661	-691.816	-1788.5	801.4895
15.0	initial	89.14785	-262.944	-863.594	335.0069
	optimal	179.1871	-528.517	-1763.46	720.265

Beam Geometry Rectangular, *Modifier* DOP *Location* Station 5 cm before Joint.

$k=0$ % DOP		$Re\{G_{w,0}\}$	$Re\{G_{j,0}\}$	$Im\{G_{w,0}\}$	$Im\{G_{j,0}\}$
2.5	initial	1195.753	-120.669	.0	.0
	optimal	1207.710	-121.876	.0	.0
5.0	initial	1195.753	85.7760	.0	.0
	optimal	1231.625	88.3493	.0	.0
7.5	initial	1195.753	-75.8661	.0	.0
	optimal	1231.625	-78.1421	.0	.0
10.0	initial	1195.753	-265.905	.0	.0
	optimal	1195.753	-265.905	.0	.0
12.5	initial	1195.753	148.2588	.0	.0
	optimal	1207.710	149.7414	.0	.0
15.0	initial	1195.753	267.9479	.0	.0
	optimal	1219.668	273.3068	.0	.0

$k=1$ % DOP		$Re\{G_{w,1}\}$	$Re\{G_{j,1}\}$	$Im\{G_{w,1}\}$	$Im\{G_{j,1}\}$
2.5	initial	89.1479	-628.76	-863.594	81.67498
	optimal	176.8693	-1247.46	-1717.17	128.0663
5.0	initial	89.1479	-612.049	-863.594	-76.773
	optimal	178.7325	-1227.1	-1732.83	-133.9
7.5	initial	89.1479	-320.744	-863.594	748.8488
	optimal	176.067	-633.469	-1706.96	1499.345
10.0	initial	89.1479	-347.877	-863.594	256.8489
	optimal	170.415	-665.002	-1662.25	439.2116
12.5	initial	89.1479	-771.77	-863.594	-374.606
	optimal	179.0802	-1550.33	-1732.95	-752.509
15.0	initial	89.1479	-863.788	-863.594	-331.456
	optimal	177.2794	-1717.73	-1722.74	-659.133

Beam Geometry Tapered *Modifier* CH, *Location* Station 5 cm before Joint.

$k=0$ % DOP		$Re\{G_{w,0}\}$	$Re\{G_{j,0}\}$	$Im\{G_{w,0}\}$	$Im\{G_{j,0}\}$
2.5	initial	1195.753	-122.929	.0	.0
	optimal	1006.205	77.34723	.0	.0
5.0	initial	1195.753	-106.246	.0	.0
	optimal	996.2434	93.26451	.0	.0
7.5	initial	1195.753	-141.959	.0	.0
	optimal	996.2434	57.5515	.0	.0
10.0	initial	1195.753	-1096.26	.0	.0
	optimal	996.2434	-896.75	.0	.0
12.5	initial	1195.753	56.20494	.0	.0
	optimal	1006.205	258.2725	.0	.0
15.0	initial	1195.753	-364.53	.0	.0
	optimal	1016.168	-168.319	.0	.0

$k=1$ % DOP		$Re\{G_{w,1}\}$	$Re\{G_{j,1}\}$	$Im\{G_{w,1}\}$	$Im\{G_{j,1}\}$
2.5	initial	89.14785	-281.602	-863.594	-263.966
	optimal	-906.089	490.3889	-1695.77	-550.805
5.0	initial	89.14785	-635.845	-863.594	-111.124
	optimal	-826.387	-250.606	-1703.57	-228.308
7.5	initial	89.14785	-563.646	-863.594	302.669
	optimal	-813.802	-106.723	-1761.62	657.0184
10.0	initial	89.14785	-372.54	-863.594	239.3359
	optimal	-872.53	283.0753	-1745.16	518.4299
12.5	initial	89.14785	-704.129	-863.594	-9.73401
	optimal	-855.751	-398.81	-1722.63	-10.3557
15.0	initial	89.14785	-264.366	-863.594	242.8598
	optimal	-918.673	534.9467	-1752.96	525.9713

Beam Geometry Rectangular, *Modifier* CH, *Location* Station 5 cm before Joint.

$k = 0$ % DOP		$Re\{G_{w,0}\}$	$Re\{G_{j,0}\}$	$Im\{G_{w,0}\}$	$Im\{G_{j,0}\}$
2.5	initial	1195.753	131.604	.0	.0
	optimal	1219.668	134.236	.0	.0
5.0	initial	1195.753	297.0649	.0	.0
	optimal	1006.205	501.5411	.0	.0
7.5	initial	1195.753	124.4898	.0	.0
	optimal	1219.668	126.9796	.0	.0
10.0	initial	1195.753	-159.055	.0	.0
	optimal	1016.168	41.26454	.0	.0
12.5	initial	1195.753	-228.594	.0	.0
	optimal	1006.205	-29.3739	.0	.0
15.0	initial	1195.753	-221.528	.0	.0
	optimal	996.2434	-22.017	.0	.0

$k = 1$ % DOP		$Re\{G_{w,1}\}$	$Re\{G_{j,1}\}$	$Im\{G_{w,1}\}$	$Im\{G_{j,1}\}$
2.5	initial	89.14785	-684.871	-863.594	-192.246
	optimal	179.1871	-1376.59	-1632.19	-397.949
5.0	initial	89.14785	-974.288	-863.594	-159.906
	optimal	-847.864	-941.18	-1751.4	-276.356
7.5	initial	89.14785	-746.585	-863.594	-545.926
	optimal	176.5127	-1478.24	-1657.24	-1140.99
10.0	initial	89.14785	-111.351	-863.594	-148.566
	optimal	-786.326	744.7073	-1635.58	-386.964
12.5	initial	89.14785	-289.863	-863.594	465.7658
	optimal	-633.423	330.344	-1317.1	557.7375
15.0	initial	89.14785	-105.088	-863.594	527.5938
	optimal	-591.475	568.9996	-1238.25	578.2613

Table 4.2: Optimal values of complex coefficients, of flexural response at a point located 5 cm before joint on a beam, with different initial populations

Seed*	I	k	$Re\{\underline{G}^*_{w,k}\}$	$Re\{\underline{G}_{j,k}\}$	$Im\{\underline{G}^*_{w,k}\}$	$Im\{\underline{G}_{j,k}\}$
0.108	71.0821	0	1207.71	-202.27	.0	.0
		1	191.6678	114.1872	-1911.14	3183.371
		2	-1219.26	685.8073	38.63196	874.1748
		3	-235.138	385.143	686.3358	-59.2043
		4	-28.439	186.6327	472.2038	-364.363
		5	-20.2839	194.2424	89.14599	-137.176
		6	146.6164	32.65966	-26.2645	44.34067
		7	73.11405	-61.4362	67.94595	36.74568
		8	40.22677	-82.9547	-8.25962	8.42146
		9	54.80875	-57.0925	-13.6086	65.62904
0.382	69.8793	0	1207.71	-202.27	.0	.0
		1	184.536	109.9384	-1911.14	3183.371
		2	-1217.06	685.8073	37.87075	856.9497
		3	-256.281	393.7496	715.3951	-61.2636
		4	-34.6214	227.2052	447.2476	-328.35
		5	-19.1463	175.044	130.2366	-200.406
		6	146.8709	43.54621	-35.1557	59.12091
		7	88.68462	-74.5199	76.84083	40.69683
		8	43.57167	-91.8746	-6.43971	6.56589
		9	59.9754	-43.1423	-10.5444	49.59297
0.563	72.0807	0	1207.71	-202.27	.0	.0
		1	195.2337	116.3115	-1861.91	3140.737
		2	-1204.42	676.6223	38.63196	874.1748
		3	-245.71	389.4464	697.0418	-60.2339
		4	-29.9847	196.7758	629.5117	-446.98
		5	-24.3988	238.2857	81.485	-125.388
		6	160.1071	25.85556	15.25271	35.10303
		7	103.5782	-87.0347	80.56336	20.34841
		8	22.84211	-45.9373	-6.25306	6.37557
		9	60.45883	-56.5758	-13.4798	65.03511

* to randomise initial population of variables.

the joint-wave across the joint.

4.3.1 Joint-wave Across the Joint

The flexural responses of beams with different geometry and joint-modifier were measured 5 cm before and 5 cm after the joint. The waves corresponding to joint were separated using the optimisation technique proposed in this work. The values of ratio of amplitude of the different frequencies after and before joint are presented in Table 4.3. It is found for rectangular beams that the amplitude ratio of the second lowest frequency (750 π rad/sec) increases with the modifier content in the joint. The amplitude ratio of the magnitude of optimal coefficients of acceleration responses (for rectangular beam with different modifier) before and after the joint is plotted in Figure 12. This ratio increases monotonically and its sensitivity increases with reducing frequencies as seen in this figure. The rate of increase of this ratio with CH is more than that with DOP, i.e., the ratio is more sensitive to variation in CH than DOP content present in the joint. However, such phenomenon is not observed in case of tapered beam.

The ultimate joint strengths of the PMMA beams were determined using universal testing machine (UTM). The specimens were made as per ASTM standards as shown in Figure 13. The load was applied at the rate of 0.2 mm/min until failure. For specimens made the same way as the beams tested earlier, the failure strength is found to be a strong monotonically decreasing function of DOP percentage as seen in Figure 14. The comparison of this figure with Figure 6, 9 and 12 suggests that the failure strength is related to the propagation of flexural waves. This suggests that the flexural wave measurements made can be used for nondestructive evaluation of joints.

Table 4.3: Amplitude ratio of frequencies after and before joint
Beam Geometry Rectangular, *Modifier* CH.

%CH → k ↓	2.5	5.0	7.5	10.0	12.5	15.0
0	1.49957	54.61816	.16864	.42247	5.06492	.89202
1	.64869	1.19715	1.28914	1.71002	2.16937	1.71706
2	1.04276	1.40495	1.38601	.81989	1.79655	2.40609
3	1.36968	1.51672	1.27785	2.72364	2.76253	2.84228
4	.35462	.58863	.71449	.58163	.51871	.39982
5	1.12579	1.73869	2.3384	.52372	1.37100	.866411
6	1.72282	1.33933	1.90781	1.02267	1.99545	1.34699
7	.61073	.55288	1.51212	1.30646	2.50985	1.81885
8	2.52937	7.56056	3.57913	1.17582	.90856	4.69645
9	.23398	.71603	1.2318	.26086	.76367	.01681

Beam Geometry Rectangular, *Modifier* DOP.

%DOP → k ↓	2.5	5.0	7.5	10.0	12.5	15.0
0	8.37945	1.02692	25.18904	1.12096	4.71934	2.78193
1	1.6459	1.11675	1.47392	1.76103	.64114	.72070
2	1.21346	1.26305	.86724	1.43915	1.57199	1.6043
3	1.77683	.94857	1.4411	1.16134	1.09662	.68471
4	.44378	.58647	.59614	.74857	.69681	.50531
5	.99490	1.67223	5.24288	1.83276	1.43869	1.37038
6	.77558	1.19486	1.77613	1.74711	1.95533	1.89644
7	1.03833	.54391	2.69166	1.30824	.72835	1.39853
8	1.21786	1.24345	.91259	1.56971	4.07491	2.20077
9	1.24846	.72489	.94147	.80794	.82791	.89733

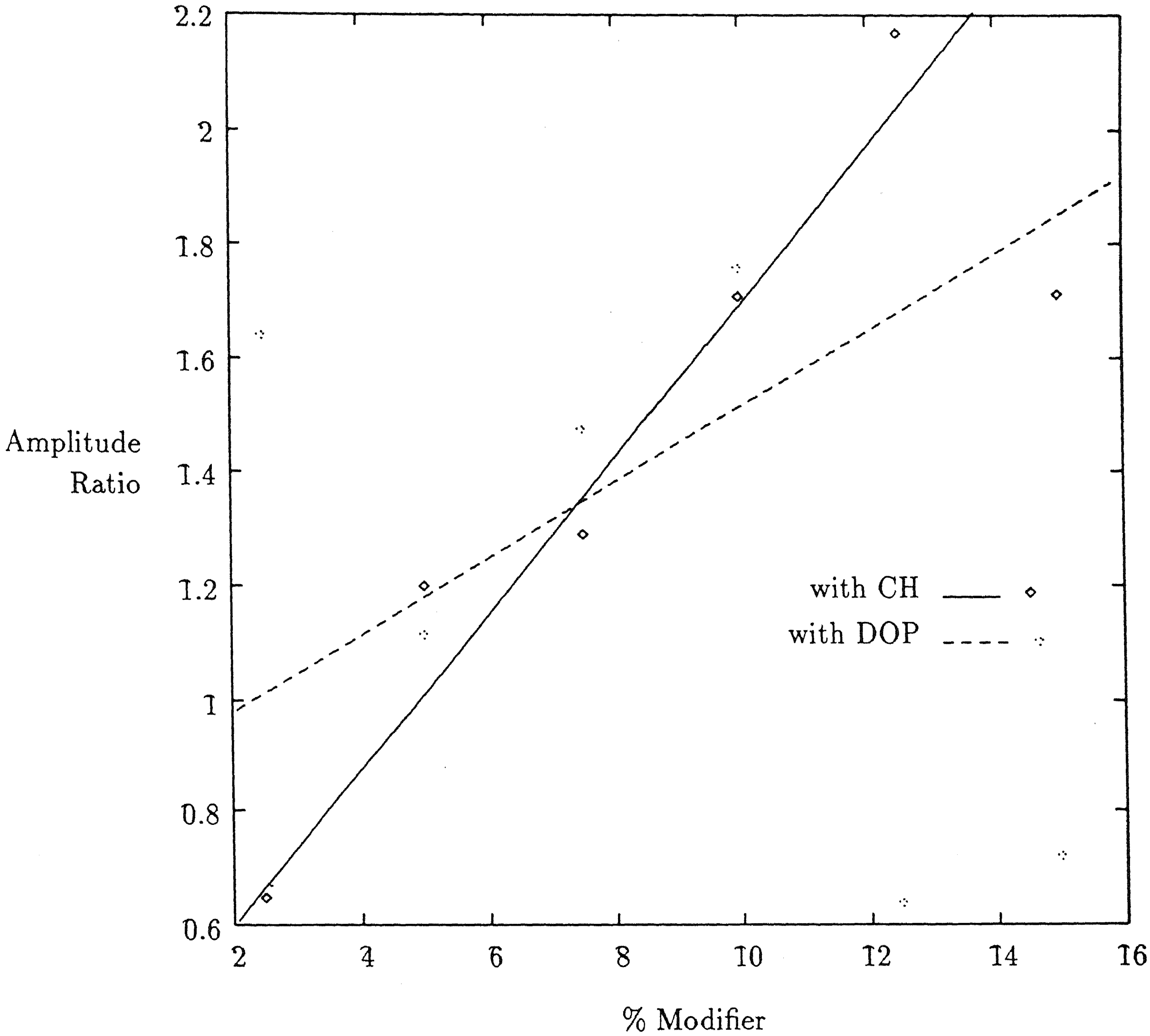


Figure 12. Amplitude ratio of joint-wave (for $k = 1$), after and before joint, in rectangular beams versus change in modifier content.

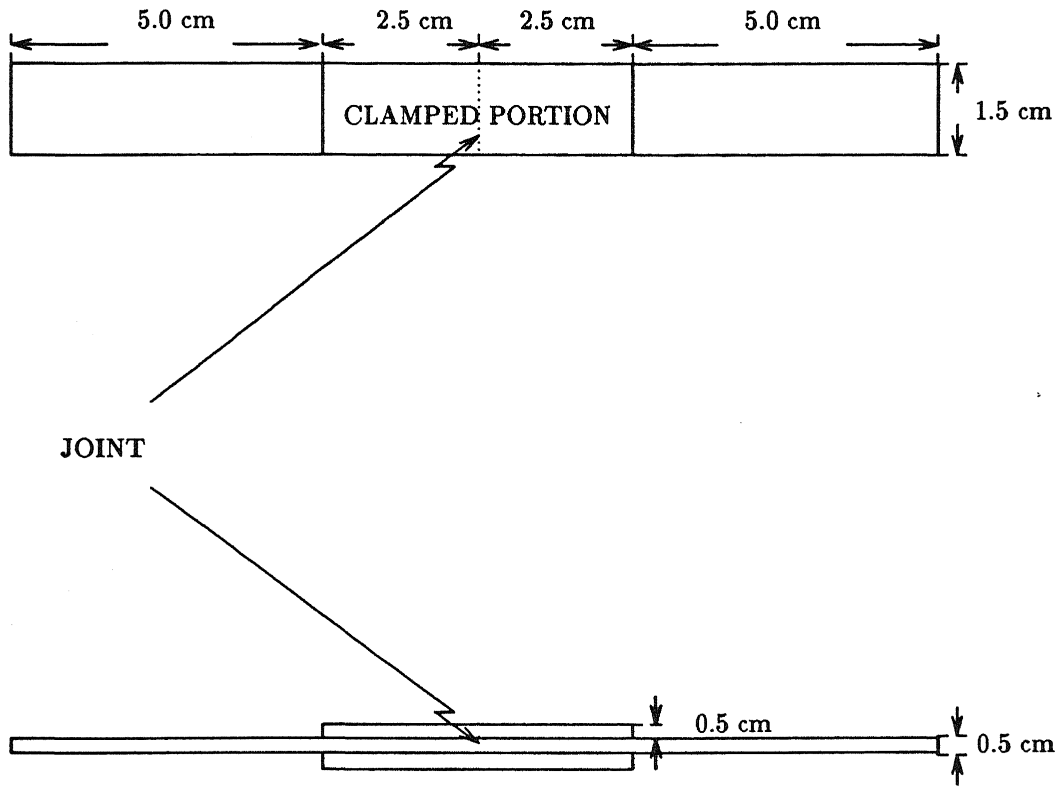


Figure 13. Specimen (of a typical beam) as per ASTM standards, tested on universal testing machine.

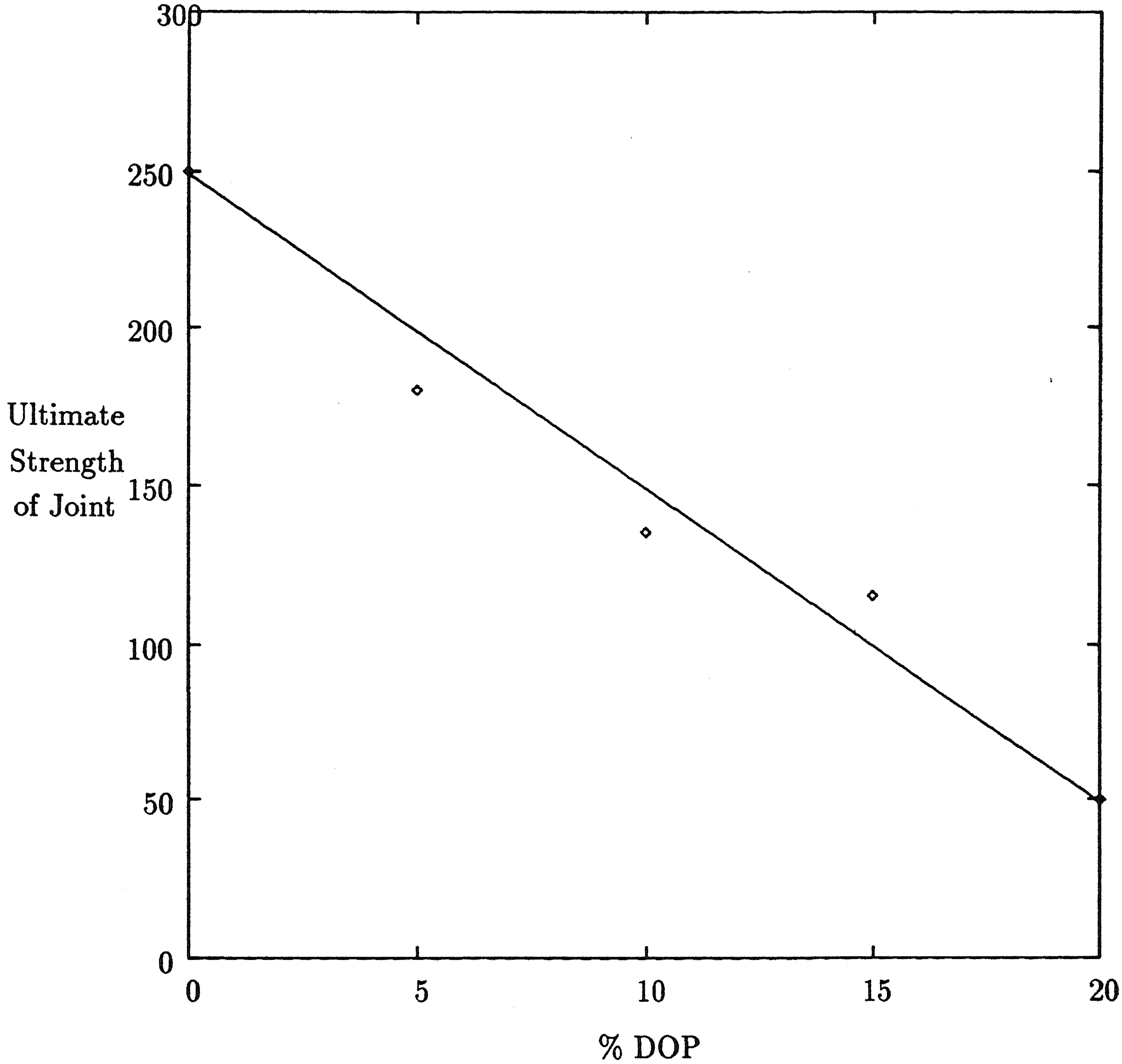


Figure 14. Ultimate strength of joint (in kgf) versus % DOP as determined from universal testing machine.

Chapter 5

Conclusions

Polymethylmethacrylate (PMMA) beams of uniform rectangular and tapered cross-section were prepared and an ultra-thin joint was introduced at the desired location using 1,2-dichloroethane (DCE). The property of the joint was varied using dioctylphthalate (DOP) plasticiser and non-solvent cyclohexane (CH) separately. Transverse flexural waves were generated by striking it with a small steel ball at its centre and the acceleration as a function of time was measured at a given location. The flexural response, so generated, appears to have a region where the information (i.e., % DOP content) concerning the nature of joint is coded. It is shown that, for rectangular beam, the FFT of this portion (section 3 of Figure 4) is linearly correlated with the % DOP for smaller frequencies. In order to carry out more controlled experiments, studies were made using a calibrated impact hammer and it was found that striking with this hammer is a multiple impact process. However, in a short time (~4 millisecond) there is always only one impact for which the acceleration versus time data at a given location on the beam was collected. The peak value of the flexural response is found to decrease with the modifier content — the rate of increase being higher with CH than with DOP irrespective of the beam geometry. The change in beam geometry does not affect the rate of decrease of peak value. With same modifier, the peak value of tapered beam is greater than that of

rectangular beam at any given amount of modifier present in the joint.

Using optimal search technique employing Genetic algorithms, the flexural wave corresponding to the joint was separated. It is shown that, for rectangular beams, the waves reflected and transmitted through the joint are related to the content of plasticiser in it. The amplitude ratio of the magnitude of the second lowest frequency (750 π rad/sec) after and before joint increase with the modifier content and is more sensitive to variation in the amount of CH in joint. The ultimate joint strength of the PMMA beams was also determined using a universal testing machine as per ASTM standards and the failure strength is found to decrease monotonically with the amount of DOP in the joint. This suggests that the technique of transverse wave measurements and analysis developed in this work can be used for non-destructive testing of joints.

References

1. R. B. Thompson and D. O. Thompson, "Past Experience in the Development of Tests for Tests for Adhesive Bond Strength", *J. Adhesion Sci. Technol.*, 5, 583 (1991).

2. D. Jian and L. Rose, "An Ultrasonic Interface Layer Model for Bond Evaluation", *J. Adhesion Sci. Technol.*, 5, 631, (1991).

3. S. E. Hannemann and V. K. Kinra, " New Technique for Ultrasonic Nondestructive Evaluation of Adhesive Joints, Part-1, Theory,", *Experimental Mechanics,* 323, Dec. (1992).

4. S. E. Hannemann and V. K. Kinra, " New Technique for Ultrasonic Nondestructive Evaluation of Adhesive Joints, Part-2, Theory,", *Experimental Mechanics,* 332, Dec. (1992).

5. A. Tiwari, G. Henneke and J. C. Duke, "Acoustic-Ultrasonic (AU) Technique for Assuming Adhesive Bond Quality", *J. of Adhesion,* 34, 1 (1991).

6. H. L. Growth, "Viscoelastic and Viscoplastic Stress Analysis of Adhesive Joints", *Int. J. Adhesion and Adhesives,*10, 207, July (1990).

7. Aleksander Pilaraski and Joseph L. Rose, "A Transverse-Wave ULtrasonic Oblique-Incidence Technique for Interfacial Weakness Detection in Adhesive Bonds", *J. App. Phys.,* 63, 2, Jan. (1988)

8. H. Ishikawa, R. Yuuki, N. Y. Chung and S. Nakano, "Mixed Mode Fracture Criteria on Adhesive Joints, Mixed Mode Fracture and Fatigue", July 15-19, Vienna, Austria.

9. M. M. Ratwani, H. P. Kan and D. D. Liu, "Time Dependent Adhesive Behaviour Effects in a Stepped Lap Joint", em American Institute of Aeronautics and Astronautics, 248, (1981).

10. M. D. Rao and M. J. Crocker, "Analytical and Experimental Study of the Vibration of Bonded Beams with a Lap Joint", *Transactions of ASME,* 112, 444, Oct. (1992).

11. S. A. Hashim, M. J. Carvlig and T. E. Winkle, "Design and Assessment Methodologies for Adhesively Bonded Structural Connections", *Int. J. Adhesion and Adhesives,*10139, July (1991).

12. D. A. Bigwood and A. D. Crocombe, "Nonlinear Adhesive Bonded Joints Design Analysis", *Int. J. Adhesion and Adhesives,*10, 31, Jan. (1990).

13. D. A. Bigwood and A. D. Crocombe, "Elastic Analysis and Engineering Design Formulae for Bonded Joints", *Int. J. Adhesion and Adhesives,*10, 339, Jan. (1990).

14. H. L. Growth and J. Brottare, "Evaluation of Singular Intensity Factors in Elastic-Plastic Materials", *Journal of Testing and Evaluation,* 291, May (1988).

15. A. Needleman, "Analysis of Interfacial Failure", *Appl. Mech. Rev.,* 43, 5274, Part-2, May (1990).

16. Z. Suo, "Failure of Brittle Adhesive Joints", *Appl. Mech. Rev.,* 5276, Part-2, May (1990).

17. Steen Krenk, "Energy Release Rate of Symmetric Adhesive Joints", *Engineering Fracture Mechanics,* 43, 549 (1992).

18. T. Kawa, H. Ishikawa and K. Muto, "The Strength of Joints Combining Adhesives with Bolts", *ISME, INT. J. Series,* 35, 38, (1992).

19. K. C. Kairouz and F. L. Mathews, "Mechanism of Failure in Bonded CFRP Single Loop Joints with Different Stacking Sequences", Proceedings of the Inst. of Mech. Engg. 47, (1990).

20. J. P. Jeandrau, "Intrinsic Mechanical Characterization of Structural Adhesives", *Int. J. Adhesion and Adhesives,*6, 229, Oct. (1986).

21. Robert D. Adams, "Strength Prediction for Lap Joints, Especially with Composite Adherends", *Int. J. Adhesion and Adhesives,*6, 229, Oct. (1986).

22. A. A. Khalil and M. R. Bayoumi. "Effect of Loading Rate on Fracture Toughness of Bonded Joints", *Int. J. Adhesion and Adhesives,*11, 25, Jan. (1991).

23. F. Edde and Y. Verreman, "On the Fracture Parameters in a Clamped Cracked Lap Shear Adhesive Joint", *Int. J. Adhesion and Adhesives,*12, 43, Jan. (1993).

24. T. Hattori, S. Sakate and G. Murakami, "A Stress Singularity Parameter Approach for Evaluating the Interfacial Reliability of Plastic Encapsulated LSI Devices", *J. of Electronic Packing,* 111, 243, Dec. (1989).

25. R. Davis and A. A. Khalil, "design and Analysis of Bonded Double Containment Corner Joints", *Int. J. Adhesion and Adhesives,*10, 25, Jan. (1990).

26. T. S. Ramamurthy and A. K. Rao, "Analysis of Bonded Joints with Arbitrary Adherend Shapes and Adhesive Non-Linearities", *Journal of Aero. Soc. of India,* 36, 29, (1983).

27. Du Chen and Shun Cheng, "An Analysis of Adhesive Bonded Single Lap Joints", *J. of Appl. Mech.,* 105, 109, March (1983).

28. J. P. Jeandrau, "Analysis and Design of Adhesive Bonded Structural Joints : New Tools for Engineers", *Mechanical Behaviour of Adhesive Joints*, 493, (1987).

29. G. Fernlund and J. K. Spelt, "Analytical Method for Calculating Adhesive Joints Fracture Parameters", *Engineering Fracture Mechanics,* 40, 119, (1991).

30. Xiaying Mu and Hurang Hu, "A Finite Element Elastic-Plastic-Creep Analysis of Materials with Temperature Dependent Properties", *Computers and Structures,* 30, 953 (1988).

31. H. L. Growth, "Calculation of Stress in Bonded Joints Using the Substructuring Technique", *Int. J. Adhesion and Adhesives,*6, 31, Jan. (1986).

32. Y. Yamada, N. Yosnimura and T. Sakurai, "Plastic Stress Strain Matrix and its Application for the Solution of Elastic-Plastic Problems by the Finite Element Method", *Int. J. Adhesion and Adhesives,*10, 343, (1968).

33. J. Had, Y. Rabah, S. S. Aivazzadehs and G. Verchery, "Edge Effects Analysis and Influence of Defects in Adhesively Bonded Composite Joints Using Interface Finite Elements", ASTM, (1986).

34. S. S. Aivazzadeh, B. Bichara, A. Ghazal and G. Verchery, "Special Mixed Finite Elements for Interfacial Stress Analysis in Adhesive Joints, Adhesively Bonded Joint Testing, Analysis and Design", ASTM, (1981).

35. S. S. Aivazzadeh, M. Babi, G. Verchery, "Assessment and Comparison of Classical, Mixed and Interface Elements for the Stress Analysis in Adhesive Joints", *Mechanical Behaviour of Adhesive Joints,* 537, (1987).

36. W. . Johnson, "Stress Analysis of the Cracked-Lap-Shear Specimen : An ASTM Round-Robin, *J. of Testing and Evaluation,* TTEVA, 15, 302 (1987).

37. T. Sawa, K., Temma and Y. Isunoda, "Axisymmetric Stress Analysis of Adhesive Butt Joints of Dissimilar Solid Cylinders Subjected to External Tensile Loads", *Int. J. Adhesion and Adhesives,*9, 161, July (1989).

38. K. Temma, T. Sawa and A. Iwata, "Two Dimensional Stress Analysis of Adhesive Butt Joints Subjected to Cleavage Loads", *Int. J. Adhesion and Adhesives,*10, 285, Oct. (1990).

39. P. Czarnocki and K. Piekarski, "Non-Linear Numerical Stress Analysis of a Symmetric Adhesive Bonded Lap Joints", *Int. J. Adhesion and Adhesives,*6, 157, July (1986).

40. K. Temma, T. Sawa and Y. Tsunoda, "Three Dimensional Stress Analysis of Adhesive Butt Joints with Disbonded Areas and Spew Fillets", *Int. J. Adhesion and Adhesives,*10, 294 (1990).

41. H. L. Growth and P. Nordlund, "Shape Optimization of Bonded Joints", *Int. J. Adhesion and Adhesives,*11, 204, Oct. (1991).

42. S. Marcolefas, V. Kostopoulos and S. A. Paipetis, "Nonlinear Analysis of a Metal to Composite Scorf Joint", *Int. J. of Mech. Sci.*, 33, 961, (1991).

43. J. P. Jeandrau, "Analysis and Design Data for Adhesively Bonded Joints", *Int. J. Adhesion and Adhesives,*11, 71, Apr. (1991).

44. K. Temma, T. Sawa and A. Iwata, "Two Dmensional Stress Analysis of Adhesive Butt Joints subjected to Cleavage Loads", *Int. J. Adhesion and Adhesives,*298, Oct. (1990).

45. Yaowu Liu and Kuan Chen, "A Two Dimensional Mesh-Generator and Variable Order Triangular and Rectangular Elements", *Computers and Structures,* 29, 161, 1033 (1988).

46. T. Sawa, K. Temma and H. Ishikawa, "Three Dimensional Stress Analysis of Adhesive Butt Joints of Solid Cylinders Subjected to External Tensile Loads", *Int. J. Adhesion and Adhesives,*31, 33, (1989).

47. Du Chen and Shun Cheng, "Stress Disribution in Plane Scarf and Butt Joints", Transactions of ASME, 57, Mar. (1990). *J. of Appl. Mech.,* 57, 78, Mar. (1990).

48. E. C. Titchmarsh, *Introduction to the theory of fourier integrals*, 2nd edition, Oxford University Press, (1948).

49. J. H. Holland, *Adaptation in natural and artificial systems* Ann Arbor: University of Michigan Press (1975).

50. D. E. Goldberg, *Genetic algorithms in search, optimization, and machine learning*, Reading, MA: Addison-Wesley (1989).

51. D. E. Goldberg and K. Deb, "A Comparative Analysis of Selection Schemes Used in Genetic Algorithms", 69-93, *Foundations of Genetic Algorithms,* edited by G. Rawlins (San Mateo, CA, Morgan Kaufman, 1991).

Appendix A

Code to Interface FFT Analyser with PC

```
5    CLS : LOCATE 4,7,1,0,31
10   PRINT "P R O G R A M   T O   T R A N S F E R   T I M E   D A T A"
20   REM
30   REM ....decl.bas
40   CLEAR , 60000!: IBINIT1 = 60000!: IBINIT2 = IBINIT1 + 3: BLOAD
         "bib.m", IBINIT1
50   CALL IBINIT1(IBFIND, IBTRG, IBCLR, IBPCT, IBSIC, IBLOC, IBPPC,
         IBBNA, IBONL, BRSC, IBSRE, IBRSV, IBPAD, IBSAD, IBIST, IBD
 IBEOS, IBTMO, IBEOT, IBRDF, IBWRTF, IBTRAP)
60   CALL IBINIT2(IBGTS, IBCAC, IBWAIT, IBPOKE, IBWRT, IBWRTA, IBCMD,
     IBCMDA, IBRD, IBRDA, IBSTOP, IBRPP, IBRSP, IBDIAG, IBXTRC, IBRDI,
     IBWRTI, IBRDIA, IBWRTIA, IBSTA%, IBERR%, IBCNT%)
80   REM ....assigning unique identifier to device and storing in DVM%
110      BDNAME$ = "KIK"
120      CALL IBFIND(BDNAME$, DVM%)
130  REM
140  REM .... checking for error on IBFIND call.
```

```
160       IF DVM% < 0 THEN GOSUB 1160
170  REM
172  REM .... clearing device.
175       CALL IBCLR(DVM%)
180  REM
190  REM  Check for an error on each GPIB call to be safe.
200  REM
210  REM ... setting sensitivity range to +10dBV.
220  WRT$ = " FRQ15 SNA8 SNB8 TDM1 TLN8 TGS0 " : CALL IBWRT(DVM%,WRT$)
230  WRT$ = " VDS1 VUL1 VAB1 VUL0 VAB0 TPV0.00025 TRP0 TRG1 "
240  CALL IBWRT(DVM%,WRT$)
250  PRINT " IBSTA% AFTER SNS7 = "; IBSTA%
260  CNT% = 4096
270  CNT2% = CNT%/2
280  DIM IAR%(5000)
340  INPUT " TYPE A KEY AFTER COLLECTING DATA ->";JUNK$
342  REM -----------------------------------------------------
344  INPUT " TYPE  0 TO STORE A CHANNEL DATA ->";CHEK%
346  IF CHEK% = 0 THEN 350 ELSE 390
350  FOR J = 0 TO CNT2%: IAR%(J) = 0: NEXT J
355  WRT$ = " THL1 TAR 00-1023? ": CALL IBWRT(DVM%,WRT$)
360  PRINT " IBSTAT% AFTER TAR? = ";IBSTA%
365  IF IBSTA% < 0 THEN GOSUB 830
370  CALL IBRDI(DVM%,IAR%(0),CNT%)
375  PRINT " NO. OF BYTES AFTER IBRDI = ";IBCNT%
380  GOSUB 950
390  REM -----------------------------------------------------
```

```
400  INPUT " TYPE 1 TO STORE B CHANNEL DATA ->"; CHEK%
410  IF CHEK% = 1 THEN 420 ELSE 590
420  FOR J = 0 TO CNT%: IAR%(J) = 0: NEXT J
430  WRT$ = " THL1 TBR 00-1023? ": CALL IBWRT(DVM%,WRT$)
440  PRINT " IBSTA% AFTER TAR = ";IBSTA%
445  IF IBSTA% = 0 THEN GOSUB 830
450  CALL IBRDI(DVM%,IAR%(0),CNT%)
455  PRINT " NO. OF BYTES AFTER IBRDI = ";IBCNT%
460  GOSUB 950
590  REM ----------data massages-----------
600  REM
610  REM -----releasing fft analyser-------------
630  V% = 1: CALL IBONL(DVM%,V%)
640  REM
650  IF IBSTA% = 0 THEN GOSUB 880
660  REM
665  INPUT " DO YOU WISH TO CONTINUE (Y/N)?";ANS$
666  IF ANS$ = "Y" THEN GOTO 5
670  END
680  REM -----END MAIN PROGRAM--------
820  PRINT "GPIB ERROR 1": RETURN
830  PRINT "GPIB ERROR 2": RETURN
840  PRINT "GPIB ERROR 3": RETURN
850  PRINT "GPIB ERROE 4": RETURN
860  PRINT "GPIB ERROR 5": RETURN
870  PRINT "GPIB ERROR 6": RETURN
880  PRINT "GPIB ERROR 7": RETURN
```

```
920  PRINT "DVM ERROR": RETURN
930  REM
950  REM -----------------processing data----------------------
970  INPUT " Please give the name of output file ->";F$
980  CLS : LOCATE 12,20, 1,0,31: PRINT " WRITING DATA. PLEASE WAIT..."
990  OPEN F$ FOR OUTPUT AS #1
1000 PRINT "CNT%=";CNT%
1010 FOR J = 0 TO (CNT2% -2) STEP 2
1020 BYTE1% = IAR%(J) AND 255
1030 BYTE2% = (IAR%(J) AND 32512) \ 256
1050 REM ----THE LEFT=MOST BYTE, BYTE2%, IS THE SIGN-BYTE------
1060 IF IAR%(J) < 0 THEN BYTE2% = BYTE2% +128
1080 BYTE3% = IAR%(J+1) AND 255
1090 BYTE4% = (IAR%(J+1) AND 32512) \ 256
1110 REM ------THE LEFT-MOST BYTE, BYTE4%, IS THE SIGN-BYTE------
1120 IF IAR%(J+1) < 0 THEN BYTE4% = BYTE4% + 128
1150 S% = (BYTE1% AND 128) \ 128
1160 N1% = BYTE1% AND 127
1170 L7% =(BYTE2% AND 128) \ 128
1180 EE1 = N1% * 2
1190 EE2 = L7%
1200 EE = EE1 + EE2
1220 N2% = BYTE2% AND 127
1230 F1 = N2% / 128
1240 F2 = BYTE3% / 32768!
1250 F3 = BYTE4% / 8388608!
1260 F = F1 + F2 + F3
```

```
1270 IF (S% = 0 AND EE = 0 AND F = 0!) THEN 1280 ELSE 1300
1280 VOLT = 0
1290 GOTO 1350
1300 EX = EE
1310 TT = 2 ^ (EX - 127)
1330 VOLT = (1! + F) * TT
1340 IF S% <> 0 THEN VOLT = -VOLT
1390 PRINT VOLT
1400 PRINT#1,VOLT
1410 NEXT J
1420 CLOSE #1
1430 RETURN
```

Appendix B

The Optimisation Code

```
#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define LINELENGTH 80     /*  width of printout */

#define BITS_PER_BYTE 8  /* number of bits per byte on this machine */

#define UINTSIZE (BITS_PER_BYTE*sizeof(unsigned))

/*-------------------------------------------------------------------*/

#define npt 1024 /* no. of data points */

#define nbit 6  /* no. of bits per variable */

#define nfq 10  /* no. of frequencies */

/*-------------------------------------------------------------------*/


/* file pointers */

FILE *outfp, *infp;

/* Global structures and variables */

struct individual

{

 unsigned *chrom;/*chromosome string for the individual*/
```

```
 double    fitness,a[nfq][6];      /* fitness of the individual */
 int       xsite;         /*  crossover site at mating */
 int       parent[2];    /* who the parents of offspring were */
 /* dynamically allocated, application-specific data structure */
};
struct bestever
{
 unsigned *chrom; /* chromosome string for the best-ever individual */
 double    fitness,a[nfq][6]; /* fitness of the best-ever individual */
 int       node;        /* node from whence came best-ever */
 int       generation; /* generation which produced it */
};

struct individual *oldpop,*newpop;
struct bestever bestfit;
double sumfitness,max,avg,min;
double ace[npt],b[nfq][6];  /* accn & frequency components */
float  pcross,pmutation;
int    numfiles,popsize,lchrom,chromsize,gen,maxgen,run;
int    maxruns,printstrings = 1,nmutation,ncross;


main(argc,argv)
int argc;
char *argv[];
{
    struct individual *temp;
```

```
FILE   *fopen();
void   *malloc();
int    i,j;
double pow();
/* determine input and output from program args */
numfiles = argc - 1;
switch(numfiles)
{
    case 0:
        infp = stdin;
        outfp = stdout;
        break;
    case 1:
        if((infp = fopen(argv[1],"r")) == NULL)
        {
            fprintf(stderr,"Input file %s not found\n",argv[1]);
            exit(-1);
        }
        outfp = stdout;
        break;
    case 2:
        if((infp = fopen(argv[1],"r")) == NULL)
        {
            fprintf(stderr,"Cannot open input file %s\n",argv[1]);
            exit(-1);
        }
        if((outfp = fopen(argv[2],"w")) == NULL)
```

```
        {
            fprintf(stderr,"Cannot open output file %s\n",argv[2]);
            exit(-1);
        }
        break;
    default:
        fprintf(stderr,"Usage : sga [input file] [output file]\n");
        exit(-1);
}

if(numfiles == 0) fprintf(outfp," accn data->");
for (i=0; i<npt; i++) fscanf(infp,"%lf",&ace[i]);
/* for(i=0;i<npt;i++)fprintf(outfp,"%f",ace[i]);
fprintf(outfp,"\n");
fprintf(outfp,"nfq= %d",nfq);*/
for (i=0; i<nfq; i++) for(j=0; j<6; j++)
fscanf(infp,"%lf",&b[i][j]);
if(numfiles == 0)
    fprintf(outfp," Number of GA runs to be performed-> ");
fscanf(infp,"%d",&maxruns);
for(run=1; run<=maxruns; run++)
{
    /* Set things up */
    initialize();
    for(gen=0; gen<maxgen; gen++)
    {
        fprintf(outfp,"\nRUN %d of %d: GENERATION %d->%d\n",
```

```
                            run,maxruns,gen,maxgen);

            /* create a new generation */
            generation();



            /* compute fitness statistics on new populations */
            statistics(newpop);

            /* report results for new generation */
            report();

            /* advance the generation */
            temp = oldpop;
            oldpop = newpop;
            newpop = temp;
        }
        freeall();
    }

}


double decodedvalue(chrom,str,stp)
int str,stp;
unsigned *chrom;
{
```

```
    int k,kstart,j,start,stop,tp,bitpos=0,mask=1,chromlen;
    double pow(), bitpow, value=0.0,temp;

    temp=(double)str/(double)UINTSIZE;
    kstart=(int)temp;
    temp=(double)stp/(double)UINTSIZE;
    chromlen=(int)(temp+1);
    for(k = kstart; k < chromlen; k++)
    {
if(k == (chromlen-1)) stop = stp-(k*UINTSIZE);
        else
    stop = UINTSIZE;
/* loop thru bits in current byte */
tp = chrom[k];
if( (k == kstart) && (k != 0) ) start = str-k*UINTSIZE;
if( k == 0 ) start = str;
if ((k != kstart) && (k != 0)) start = 0;
for(j = start; j < stop; j++)
           {
    /* test for current bit 0 or 1 */
    if((tp&mask) == 1)
       {
bitpow = pow(2.0,(double) bitpos);
value += bitpow;
       }
    tp = tp>>1;
    bitpos = bitpos+1;
```

```
    }
     }
     return(value);
}


objfunc(critter)
struct individual *critter;
{


     int i,j,k=0;
     double a1[nfq][6],fwd[nfq][2];
     double pi=3.14159,err,x,y,sum,decodedvalue();

     for(i=0;i<nfq;i++) for(j=0;j<6;j++) a1[i][j]=0;

     for(i=0;i<nfq;i++) for(j=0;j<6;j++)
     {
      a1[i][j]=b[i][j]*(1+decodedvalue(critter->chrom,k,k+nbit)/100);
      critter->a[i][j] = a1[i][j];
      k=k+nbit;
     }

     for(i=0;i<nfq;i++)
     {
          fwd[i][0]=a1[i][0]+a1[i][1]+a1[i][2];
          fwd[i][1]=a1[i][3]+a1[i][4]+a1[i][5];
     }
```

```
    err = 0;
    for(i=0;i<npt;i++)
     {
      sum=0;
      for(j=0;j<nfq;j++)
       {
x = (fwd[j][0]*cos(2*pi*i*j/npt))/npt;
  y = -(fwd[j][1]*sin(2*pi*i*j/npt))/npt;
sum = sum+x+y;
       }
      err = err+(ace[i]-sum)*(ace[i]-sum);
     }

     fprintf(outfp,"err=%f\n\n",err);
     critter->fitness = pow(1E3/err,(double)2.0);
}

double guessfit()
{
  int i,j;
  double err,sum,x,y,pi=3.14159;

    err = 0;
    for(i=0;i<npt;i++)
     {
      sum=0;
```

```
      for(j=0;j<nfq;j++)
       {
x = (b[j][0]*cos(2*pi*i*j/npt))/npt;
  y = -(b[j][1]*sin(2*pi*i*j/npt))/npt;
sum = sum+x+y;
       }
      err = err+(ace[i]-sum)*(ace[i]-sum);
     }
    return(err);
}

curve()
{
  int i,j;
  double sum,x,y,pi=3.14159;

    fprintf(outfp,"..Predicted Response..\n\n");
    for(i=0;i<npt;i++)
     {
      sum=0;
      for(j=0;j<nfq;j++)
       {
        x=bestfit.a[j][0]+bestfit.a[j][1]+bestfit.a[j][2];
        y=bestfit.a[j][3]+bestfit.a[j][4]+bestfit.a[j][5];
x = (x*cos(2*pi*i*j/npt))/npt;
  y = -(y*sin(2*pi*i*j/npt))/npt;
sum = sum+x+y;
```

```
        }
       fprintf(outfp,"%d  %f\n",i,sum);
      }
}

getarray(sarray,size)
int *sarray,size;
{
    int i,*temparray,temp,pos;

    temparray = (int *)malloc(popsize * sizeof(int));
    for (i=0; i<popsize; i++) temparray[i] = i;
    for (i=0; i<popsize-1; i++) {
pos = rnd(i,popsize);
temp = temparray[pos];
temparray[pos] = temparray[i];
temparray[i] = temp;
    }
    for (i=0; i<size; i++) sarray[i] = temparray[i];
    free(temparray);
}


generation()
{

  int mate1, mate2, jcross, j = 0;
```

```
/* do preselection actions necessary before generation */
preselect();

/* select, crossover, and mutation */
do
  {
    /* pick a pair of mates */
    mate1 = select();
    mate2 = select();

    /* Crossover and mutation */
    jcross = crossover(oldpop[mate1].chrom, oldpop[mate2].chrom,
newpop[j].chrom, newpop[j+1].chrom);
    mutation(newpop[j].chrom);
    mutation(newpop[j+1].chrom);

    /* Decode string, evaluate fitness, & record */
    /* parentage date on both children */
    objfunc(&(newpop[j]));
    newpop[j].parent[0] = mate1+1;
    newpop[j].xsite = jcross;
    newpop[j].parent[1] = mate2+1;
    objfunc(&(newpop[j+1]));
    newpop[j+1].parent[0] = mate1+1;
    newpop[j+1].xsite = jcross;
    newpop[j+1].parent[1] = mate2+1;
```

```
      /* Increment population index */
      j = j + 2;
    }
  while(j < (popsize-1));

}


initialize()
/* Initialization Coordinator */
{
 int i,j;

    /* get basic problem values from input file */
    initdata();
    /* define chromosome size in terms of machine bytes, ie  */
    /* length of chromosome in bits (lchrom)/(bits-per-byte) */
    /* chromsize must be known for malloc() of chrom pointer */
    chromsize = (lchrom/UINTSIZE);
    if(lchrom%UINTSIZE) chromsize++;

    /* malloc space for global data structures */
    initmalloc();

    /* initialize random number generator */
    randomize();
```

```
    /* initialize global counters/values */
    nmutation = 0;
    ncross = 0;
    bestfit.generation = 0;
    bestfit.fitness = 0.0;
    for(i=0;i<nfq;i++) for(j=0;j<6;j++) bestfit.a[i][j] = 0;

    /* initialize the populations and report statistics */
    initpop();
    statistics(oldpop);
    initreport();
}


initdata()
/* data inquiry and setup */
{
    char  answer[2];

    if(numfiles == 0)
    {
    fprintf(outfp,"\n ...Data Entry and Initialization...\n");
        fprintf(outfp," Enter the population  size ----------> ");
    }
    fscanf(infp,"%d", &popsize);
```

```
    if(numfiles == 0)
        fprintf(outfp," Enter chromosome length --------------> ");
    fscanf(infp,"%d", &lchrom);
    if(nfq*nbit*5 > lchrom)
    {
     fprintf(outfp,"Insufficient lchrom !!\n");
     exit(-1);
    }

    if(numfiles == 0)
        fprintf(outfp," Print chromosome strings? (y/n) ------> ");
    fscanf(infp,"%s",answer);
    if(strncmp(answer,"n",1) == 0) printstrings = 0;

    if(numfiles == 0)
        fprintf(outfp," Enter maximum no. of generations -----> ");
    fscanf(infp,"%d", &maxgen);

    if(numfiles == 0)
        fprintf(outfp," Enter crossover probability ----------> ");
    fscanf(infp,"%f", &pcross);

    if(numfiles == 0)
        fprintf(outfp," Enter mutation probability -----------> ");
    fscanf(infp,"%f", &pmutation);
}
```

```
initpop()
/* Initialize a population at random */
{
    int j, j1, k, stop;
    unsigned mask = 1;

    for(j = 0; j < popsize; j++)
    {
        for(k = 0; k < chromsize; k++)
        {
            oldpop[j].chrom[k] = 0;
            if(k == (chromsize-1))
                stop = lchrom - (k*UINTSIZE);
            else
                stop = UINTSIZE;
            /* A fair coin toss */
            for(j1 = 1; j1 <= stop; j1++)
            {
               if(flip(0.5))
                    oldpop[j].chrom[k] = oldpop[j].chrom[k]|mask;
                        if (j1 != stop) oldpop[j]
      .chrom[k] = oldpop[j].chrom[k]<<1;
            }
        }
        oldpop[j].parent[0] = 0; /* Initialize parent info. */
        oldpop[j].parent[1] = 0;
```

```
        oldpop[j].xsite = 0;
        objfunc(&(oldpop[j]));  /* Evaluate initial fitness */
    }
}


initreport()
/* Initial report */
{
  void   skip();

  skip(outfp,1);
  fprintf(outfp," SGA Parameters\n");
  fprintf(outfp," ----------------------------\n");
  fprintf(outfp," Total Population size = %d\n",popsize);
  fprintf(outfp," Chromosome length = %d\n",lchrom);
  fprintf(outfp," Maximum # of generations = %d\n",maxgen);
  fprintf(outfp," Crossover probability  = %f\n",pcross);
  fprintf(outfp," Mutation probability  = %f\n",pmutation);
  skip(outfp,1);
 }

initmalloc()
/* memory allocation of space for global data structures */
{
  unsigned nbytes;
  void    *malloc();
```

```
  int j;

  /* memory for old and new populations of individuals */
  nbytes = popsize*sizeof(struct individual);
  if((oldpop = (struct individual *) malloc(nbytes)) == NULL)
    nomemory(stderr,"oldpop");

  if((newpop = (struct individual *) malloc(nbytes)) == NULL)
    nomemory(stderr,"newpop");

  /* memory for chromosome strings in populations */
  nbytes = chromsize*sizeof(unsigned);
  for(j = 0; j < popsize; j++)
    {
      if((oldpop[j].chrom = (unsigned *) malloc(nbytes)) == NULL)
nomemory(stderr,"oldpop chromosomes");

      if((newpop[j].chrom = (unsigned *) malloc(nbytes)) == NULL)
nomemory(stderr,"newpop chromosomes");
    }
  if((bestfit.chrom = (unsigned *) malloc(nbytes)) == NULL)
      nomemory(stderr,"bestfit chromosomes");

}


freeall()
```

```
     /* A routine to free all the space dynamically allocated
in initspace() */
{
  int i;

  for(i = 0; i < popsize; i++)
    {
      free(oldpop[i].chrom);
      free(newpop[i].chrom);
    }
  free(oldpop);
  free(newpop);
  free(bestfit.chrom);
}


nomemory(string)
     char *string;
{
  fprintf(outfp,"malloc: out of memory making %s!!\n",string);
  exit(-1);
}


mutation(child)
unsigned *child;
/* Mutate an allele w/ pmutation, count # of mutations */
```

```
{
    int j, k, stop;
    unsigned mask, temp = 1;

    for(k = 0; k < chromsize; k++)
    {
        mask = 0;
        if(k == (chromsize-1))
            stop = lchrom - ((k-1)*UINTSIZE);
        else
            stop = UINTSIZE;
        for(j = 0; j < stop; j++)
        {
            if(flip(pmutation))
            {
                mask = mask|(temp<<j);
                nmutation++;
            }
        }
        child[k] = child[k]^mask;
    }
}


int crossover (parent1, parent2, child1, child2)
unsigned *parent1, *parent2, *child1, *child2;
/* Cross 2 parent strings, place in 2 child strings */
```

```
{
    int j, jcross, k;
    unsigned mask, temp;

    /* Do crossover with probability pcross */
    if(flip(pcross))
    {
        jcross = rnd(1 ,(lchrom - 1));/* Cross between 1 and l-1 */
        ncross++;
        for(k = 1; k <= chromsize; k++)
        {
            if(jcross >= (k*UINTSIZE))
            {
                child1[k-1] = parent1[k-1];
                child2[k-1] = parent2[k-1];
            }
            else if((jcross < (k*UINTSIZE)) &&
           (jcross > ((k-1)*UINTSIZE)))
            {
                mask = 1;
                for(j = 1; j <= (jcross-1-((k-1)*UINTSIZE)); j++)
                {
                    temp = 1;
                    mask = mask<<1;
                    mask = mask|temp;
                }
             child1[k-1] = (parent1[k-1]&mask)|(parent2[k-1]&(~mask));
```

```
                child2[k-1] = (parent1[k-1]&(~mask))|(parent2[k-1]&mask);
            }
            else
            {
                child1[k-1] = parent2[k-1];
                child2[k-1] = parent1[k-1];
            }
        }
    }
    else
    {
        for(k = 0; k < chromsize; k++)
        {
            child1[k] = parent1[k];
            child2[k] = parent2[k];
        }
        jcross = 0;
    }
    return(jcross);
}

#include <math.h>

/* variables are declared static so that they cannot conflict
with names of    */
static double oldrand[55];    /* Array of 55 random numbers */
static int jrand;             /* current random number */
```

```
advance_random()
/* Create next batch of 55 random numbers */
{
    int j1;
    double new_random;

    for(j1 = 0; j1 < 24; j1++)
    {
        new_random = oldrand[j1] - oldrand[j1+31];
        if(new_random < 0.0) new_random = new_random + 1.0;
        oldrand[j1] = new_random;
    }
    for(j1 = 24; j1 < 55; j1++)
    {
        new_random = oldrand [j1] - oldrand [j1-24];
        if(new_random < 0.0) new_random = new_random + 1.0;
        oldrand[j1] = new_random;
    }
}


int flip(prob)
/* Flip a biased coin - true if heads */
float prob;
{
    float randomperc();
```

```
    if(randomperc() <= prob)
        return(1);
    else
        return(0);
}


randomize()
/* Get seed number for random and start it up */
{
    float randomseed;
    int j1;

    for(j1=0; j1<=54; j1++) oldrand[j1] = 0.0;
    jrand=0;

    if(numfiles == 0)
    {
        do
        {
         fprintf(outfp,"Enter random number seed, 0.0 to 1.0 ->");
         fscanf(infp,"%f", &randomseed);
        }
        while((randomseed < 0.0) || (randomseed > 1.0));
    }
    else
```

```
    {
        fscanf(infp,"%f", &randomseed);
    }


    warmup_random(randomseed);
}


float randomperc()
{
    jrand++;
    if(jrand >= 55)
    {
        jrand = 1;
        advance_random();
    }
    return((float) oldrand[jrand]);
}


int rnd(low, high)
/* Pick a random integer between low and high */
int low,high;
{
    int i;
    float randomperc();
```

```
    if(low >= high)
        i = low;
    else
    {
        i = (randomperc() * (high - low + 1)) + low;
        if(i > high) i = high;
    }
    return(i);
}



warmup_random(random_seed)
/* Get random off and running */
float random_seed;
{
    int j1, ii;
    double new_random, prev_random;

    oldrand[54] = random_seed;
    new_random = 0.000000001;
    prev_random = random_seed;
    for(j1 = 1 ; j1 <= 54; j1++)
    {
        ii = (21*j1)%54;
        oldrand[ii] = new_random;
        new_random = prev_random-new_random;
```

```
        if(new_random<0.0) new_random = new_random + 1.0;
        prev_random = oldrand[ii];
    }

    advance_random();
    advance_random();
    advance_random();

    jrand = 0;
}



report()
/* Write the population report */
{
    void   repchar(), skip();
    int    i,j,writepop(), writestats();
    double guessfit();
    if (gen == maxgen-1)
    {
    repchar(outfp,"-",LINELENGTH);
    skip(outfp,1);
    if(printstrings == 1)
    {
        repchar(outfp," ",((LINELENGTH-17)/2));
        fprintf(outfp,"Population Report\n");
        fprintf(outfp, "Generation %3d", gen);
```

```
    repchar(outfp," ",(LINELENGTH-28));
    fprintf(outfp, "Generation %3d\n", (gen+1));
    fprintf(outfp,"num   string ");
    repchar(outfp," ",lchrom-5);
    fprintf(outfp,"fitness    parents xsite  ");
    fprintf(outfp,"string ");
    repchar(outfp," ",lchrom-5);
    fprintf(outfp,"fitness\n");
    repchar(outfp,"-",LINELENGTH);
    skip(outfp,1);

    /* write out string info from all nodes         */
    /* in order from lowest to highest node number  */
    writepop(outfp);
    repchar(outfp,"-",LINELENGTH);
    skip(outfp,1);

}

/* write the summary statistics in global mode  */
fprintf(outfp,"Generation %d Accumulated Statistics: \n",
        gen);

fprintf(outfp,"Total Crossovers = %d, Total Mutations = %d\n",
               ncross,nmutation);
fprintf(outfp,"min = %f   max = %f   avg = %f   sum = %f\n",
             1/min,1/max,1/avg,1/sumfitness);
```

```
fprintf(outfp,"Global Best Individual so far, Generation %d:\n",
            bestfit.generation);
fprintf(outfp,"Fitness = %f: ",1000*
pow(1/bestfit.fitness,(double).5));
for (i=0; i<nfq; i++)
{
 fprintf(outfp,"\n");
 fprintf(outfp," %f  %f ",bestfit.a[i][0]+bestfit.a[i][1]+
 bestfit.a[i][2],bestfit.a[i][3]+bestfit.a[i][4]+bestfit.a[i][5]);
}
fprintf(outfp,"\n\n..Initial Guess..\n");
fprintf(outfp,"Guess fitness = %f \n",guessfit());
for (i=0; i<nfq; i++)
{
 fprintf(outfp,"\n");
 fprintf(outfp," %f  %f ",b[i][0]+b[i][1]+b[i][2],b[i][3]+
 b[i][4]+b[i][5]);
}
fprintf(outfp,"\n");
for (i=0; i<nfq; i++)
{
 fprintf(outfp,"\n");
 for(j=0; j<6; j++) fprintf(outfp," %f ",bestfit.a[i][j]);
}
fprintf(outfp,"\n\n..Initial Guess..\n");
for (i=0; i<nfq; i++)
{
```

```
     fprintf(outfp,"\n");
     for(j=0; j<6; j++) fprintf(outfp," %f ",b[i][j]);
    }
    writechrom((&bestfit)->chrom);
    skip(outfp,1);
    repchar(outfp,"-",LINELENGTH);
    skip(outfp,1);
    curve();
    }
}


writepop()
{
    struct individual *pind;
    int j;

    for(j=0; j<popsize; j++)
    {
        fprintf(outfp,"%3d)  ",j+1);

        /* Old string */
        pind = &(oldpop[j]);
        writechrom(pind->chrom);
        fprintf(outfp,"  %8f | ", 1/pind->fitness);

        /* New string */
```

```
        pind = &(newpop[j]);
        fprintf(outfp,"(%2d,%2d)   %2d   ",
        pind->parent[0], pind->parent[1], pind->xsite);
        writechrom(pind->chrom);
        fprintf(outfp,"  %8f\n", 1/pind->fitness);
    }
}


writechrom(chrom)
/* Write a chromosome as a string of ones and zeroes            */
/* note that the most significant bit of the chromosome is the  */
/* RIGHTMOST bit, not the leftmost bit, as would be expected... */
unsigned *chrom;
{
    int j, k, stop;
    unsigned mask = 1, tmp;

    for(k = 0; k < chromsize; k++)
    {
        tmp = chrom[k];
        if(k == (chromsize-1))
            stop = lchrom - (k*UINTSIZE);
        else
            stop = UINTSIZE;

        for(j = 0; j < stop; j++)
```

```
        {
            if(tmp&mask)
                fprintf(outfp,"1");
            else
                fprintf(outfp,"0");
            tmp = tmp>>1;
        }
    }
}


preselect()
{
    int j;

    sumfitness = 0;
    for(j = 0; j < popsize; j++) sumfitness += oldpop[j].fitness;
}

int select()
/* roulette-wheel selection */
{
    extern float randomperc();
    float sum, pick;
    int i;

    pick = randomperc();
```

```
    sum = 0;

    if(sumfitness != 0)
    {
        for(i = 0; (sum < pick) && (i < popsize); i++)
            sum += oldpop[i].fitness/sumfitness;
    }
    else
        i = rnd(0,popsize-1);

    return(i-1);
}



statistics(pop)
/* Calculate population statistics */
struct individual *pop;
{
    int i,j,k;

    sumfitness = 0.0;
    min = pop[0].fitness;
    max = pop[0].fitness;

    /* Loop for max, min, sumfitness */
    for(j = 0; j < popsize; j++)
    {
```

```
        sumfitness = sumfitness + pop[j].fitness;
 /* Accumulate */
        if(pop[j].fitness > max) max = pop[j].fitness;
/* New maximum */
        if(pop[j].fitness < min) min = pop[j].fitness;
/* New minimum */

        /* new global best-fit individual */
        if(pop[j].fitness > bestfit.fitness)
      {
       for (i=0; i<chromsize; i++) bestfit.chrom[i]
       = pop[j].chrom[i];
       for (i=0; i<nfq; i++) for (k=0;k<6; k++) bestfit.a[i][k] =
       pop[j].a[i][k];
       bestfit.fitness    = pop[j].fitness;
       bestfit.generation = gen;
        }
    }

    /* Calculate average */
    avg = sumfitness/popsize;
}


void repchar (outfp,ch,repcount)
/* Repeatedly write a character to stdout */
FILE *outfp;
```

```
char *ch;
int repcount;
{
    int j;

    for (j = 1; j <= repcount; j++) fprintf(outfp,"%s", ch);
}


void skip(outfp,skipcount)
/* Skip skipcount lines */
FILE *outfp;
int skipcount;
{
    int j;

    for (j = 1; j <= skipcount; j++) fprintf(outfp,"\n");
}

double pow(a,b)
double a,b;
{
   return((a==0) ? 0.0 : exp(b*log(a)));
}
```